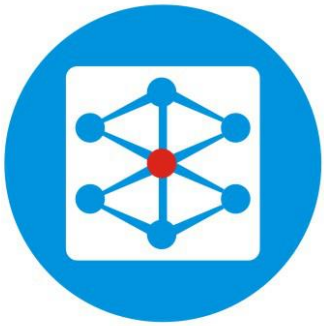


BlockCDN

The new revolution of CDN

BlockCDN WhitePaper

A blockchain-powered CDN trading platform

Contents

Version: 1.0

Abstract

The white paper describes BlockCDN is a decentralized CDN trading market and explains how to deploy Solidity to operate smart contract on Ethereum. With blockchain technology, it calls for global internet idler to share their idle equipment (e.g. personal computer, router, TV box, tablet, mobile phone) and provide the uploaded traffic to make internet fast node ubiquitous with faster access to internet and less CDN services used by bandwidth. In addition, store the data of websites in distributed way could make the source websites safer. Blockchain and smart contract guarantees idle broadband users can gain good benefits from sharing their idle equipment and uploading traffic without additional increase in investment. What`s more, the website owners who need speed-up will acquire a new type of CDN service which is 10% of price of traditional CDN, more nodes and faster.

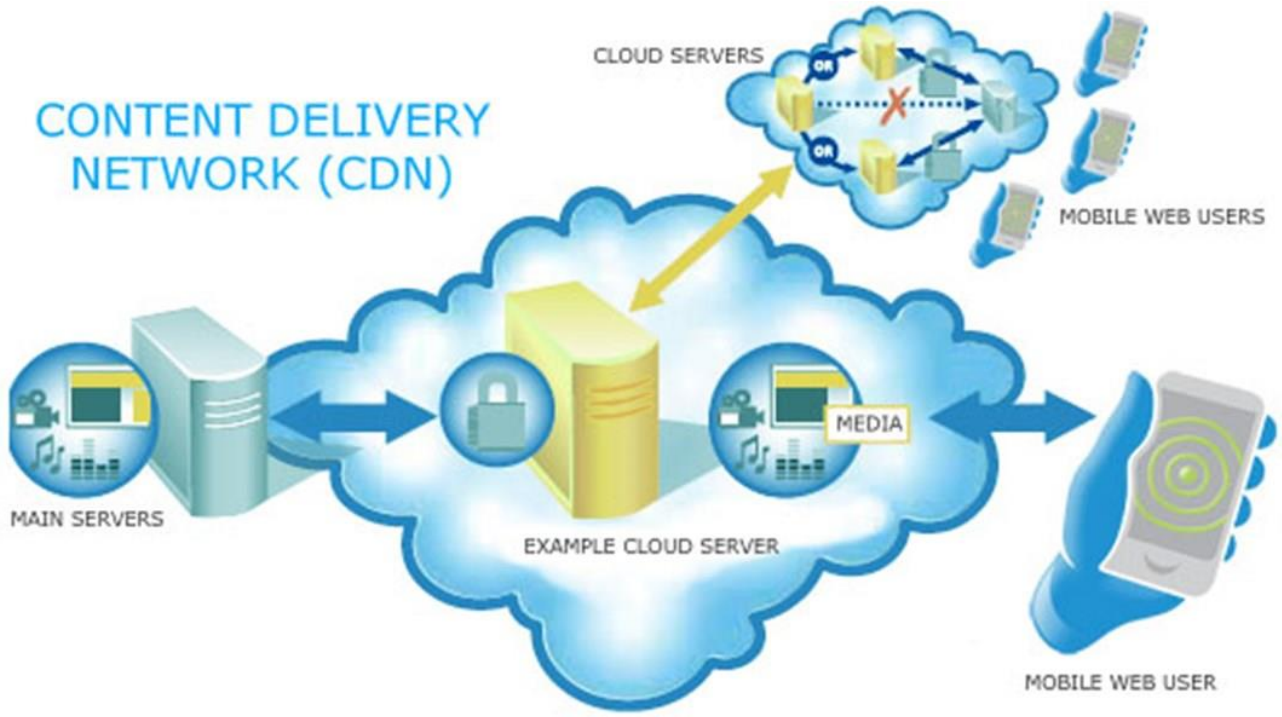
1. Concept

What is CDN?

CDN，Content Delivery Network is distributed network consisting of node server groups located in different regions and deliver its contents, such as internet websites, internet videos and online games to edge node servers near the end-users with high availability, which is an important means for reducing network congestion, increasing the speed of internet business response and improving user's business experience.

For detailed information, please refer to Wikipedia：

［1］ https://en.wikipedia.org/wiki/Content_delivery_network



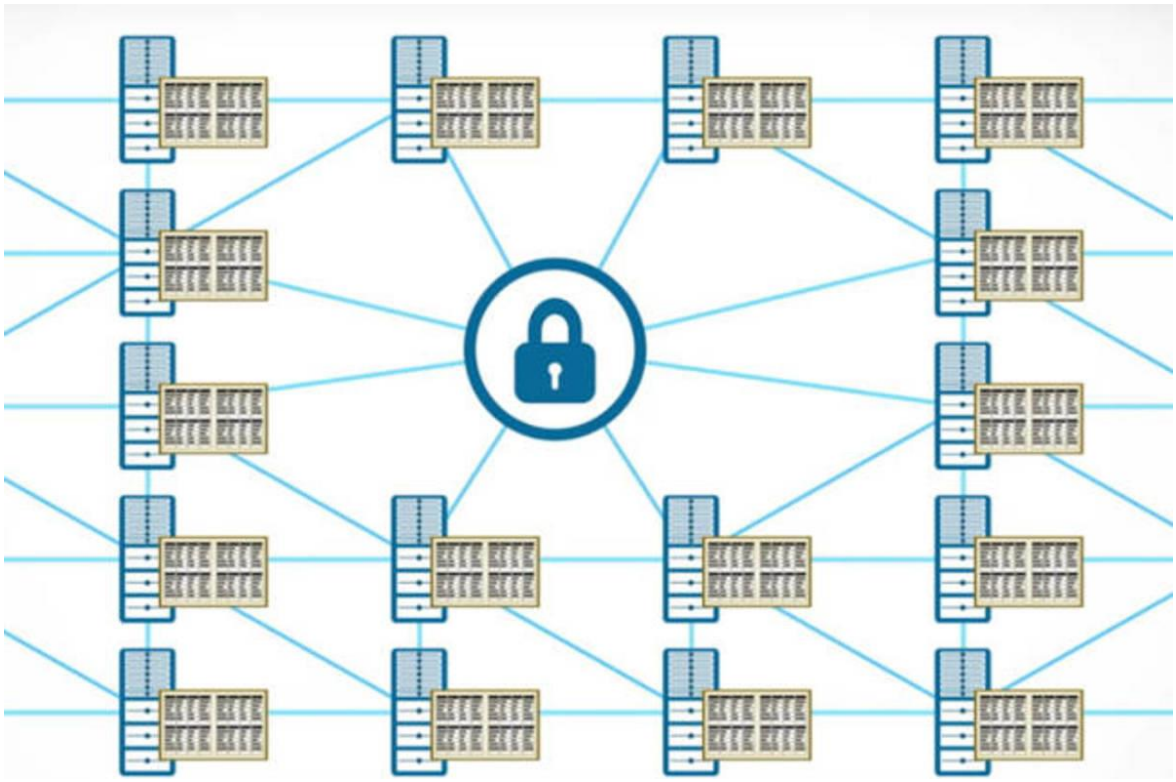
What is blockchain?

Blockchain is a technical solution, which collectively maintains a reliable database through decentralization and de-trust mode. The technical solution mainly makes any multiple nodes which are engaged in the system to produce block with the method of

cryptography and each block contains the whole information exchange data within given time, and produce data fingerprint which is used for validating the effectiveness of information and chain the next data block.

For detailed information, please refer to Wikipedia：

［2］https://en.wikipedia.org/wiki/Block_chain_(database)

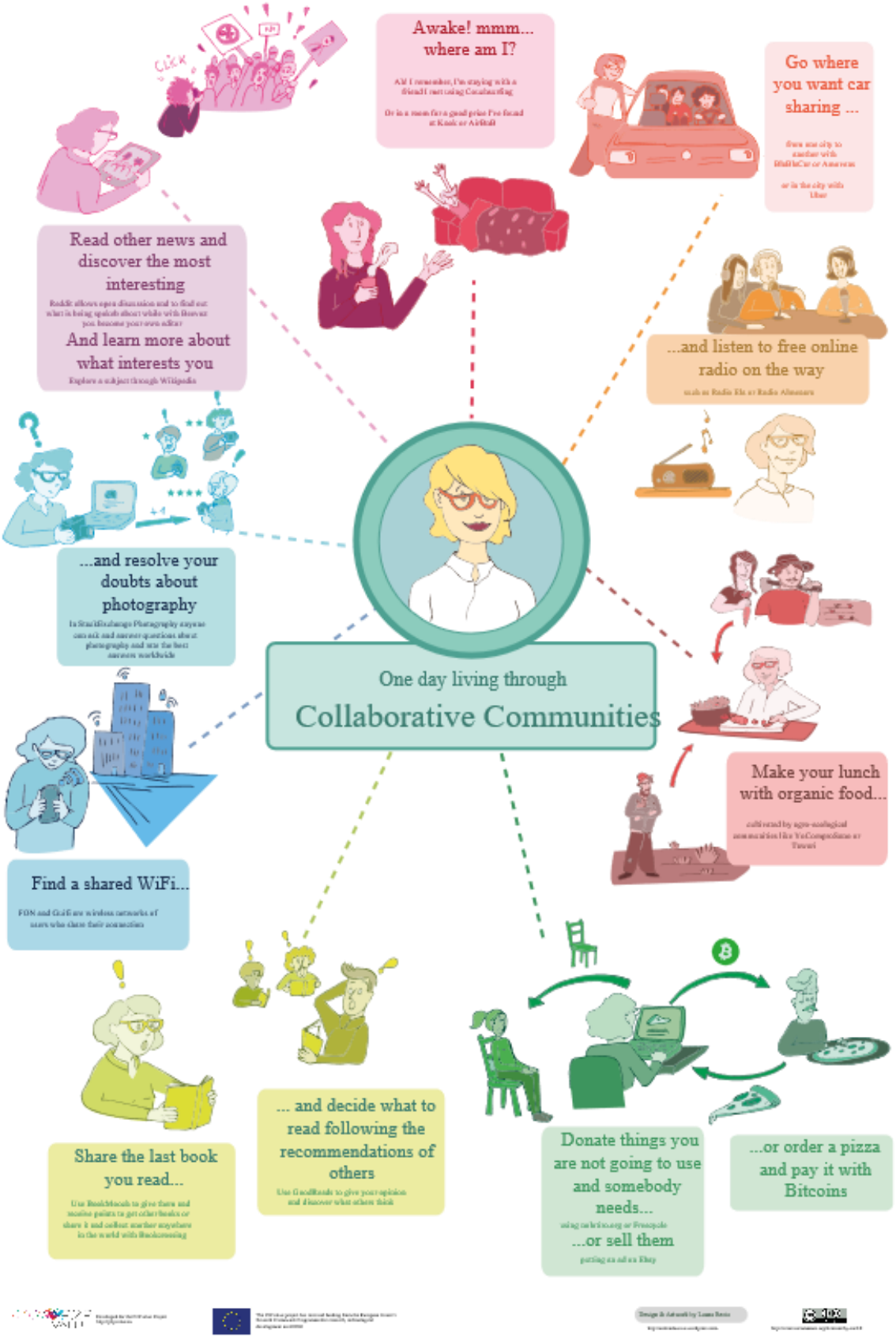


What is sharing economy?

Sharing Economy, also called peer-to-peer economy, coordination economy, collaborative consumption，is a social economic ecosystem built on sharing of human and material resources, for example, Uber.

For detailed information, please refer to Wikipedia：

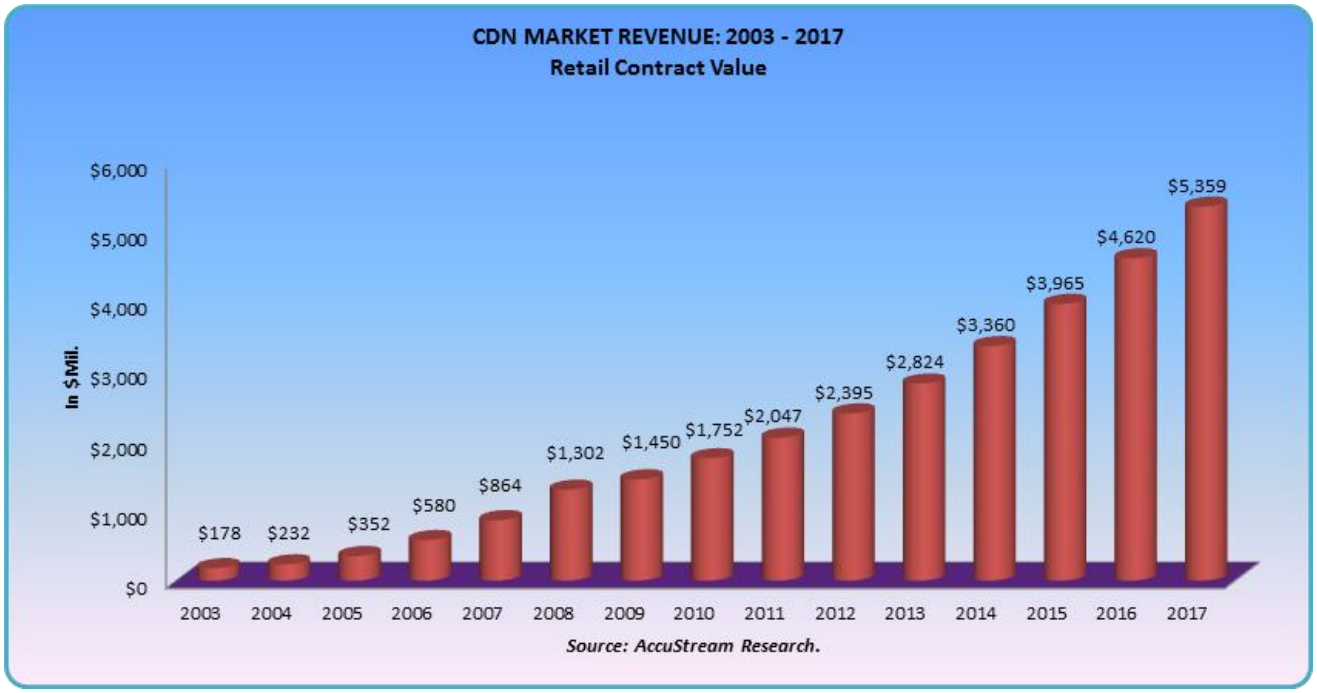
［3］https://en.wikipedia.org/wiki/Sharing_economy


Awake! mmm... where am I?
Go where you want car sharing ...
Read other news and discover the most interesting
And learn more about what interests you
...and listen to free online radio on the way
...and resolve your doubts about photography
One day living through
Collaborative Communities
Make your lunch with organic food...
Find a shared WiFi...
Share the last book you read...
... and decide what to read following the recommendations of others
Donate things you are not going to use and somebody needs...
...or sell them
...or order a pizza and pay it with Bitcoins

2. What changes will BlockCDN bring to market?

The current situation of CDN:

The global scale of construction and market of CDN is constantly expanding. According to the report data from MicroMarketMonitor released by PRWeb, the size of CDN market is estimated to increase from $3.7034 billion in 2014 to $12.1637billion in 2019 with annual compound growth rate of 26.9% in revenue.

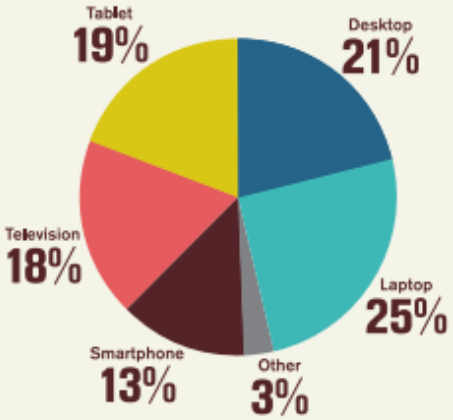


CDN has numerous advantages, such as solving the problems of ISP intercommunication and network link issue, lessening the pressure of source server,anti-attacking effectively DDOS. However, there are also disadvantages: it`s complicated to operate the caching server, require enormous investment and the number of layout is limited.

The following picture comes from：https://www.akamai.com

STATE OF CDN SERVICES

REACHING GLOBAL SCALE AND MULTIPLE SCREENS USING CONTENT DELIVERY NETWORKS



Primary Personal On-Demand Media Viewing

Desktops and **laptops** still dominate personal online media viewing, accounting for 46%, but **tablets** now outrank OTT (television) to come in **third place.**

PREMIUM content, at **59%** of media consumed, is **much higher** than **social media** (23%) and **enterprise** (13%).

Yet companies seem to focus on delivering **enterprise** (28%) at almost **4x** that of social media delivery (8%).

Premium delivery focus by companies in the industry is only **33%** or **half** of what participants say they consumed (59%).

90% of all respondents said the **quality** of the content they watch online is **important** or **very important.**

In addition, those premium movies (or cat videos) need to be delivered to **multiple screens**, which is the **number-one request** for companies in the industry from their CDN provider, followed by the need for adaptive bitrate technologies.

The Advantages of BlockCDN :

A、As BlockCDN is based on blockchain technology and makes global internet user to share their idle equipment as data caching nodes and provide data uploading service for nearby accessing users. And thus will get corresponding rewards.

With more people engaged in BlockCDN, the CDN nodes become more abundant, which will form a global distributed CDN network and revolutionize the present CDN operation mode so as to make the most of resources infinitely close to caching nodes, increase significantly the speed of access as well as effectively relief the network congestion.
The node sharers can get the money from their nodes. No matter they are walking dog, going shopping, having dinner or wathing movies.

B、With increasingly abundant in website contents and APP, the network congestion becomes more serious and user`s experience gradually declines. The expensive acceleration fee of CDN still constantly exploit the website owner's profit. Being a self-service CDN trading market,BlockCDN make the website owners and sharers enjoy free, equal and transparent trade, bring down 90% of acceleration fee and get better CDN service.

OUTBOUND DATA TRANSFERS	Akamai CDN	Verizon CDN	BlockCDN Guide price	Discount	Wages for Distributors
First 10 TB 2 /Month	$0.087 per GB	$0.17 per GB	$0.008 per GB	95% OFF	$0.0072 per GB
Next 40 TB (10 – 50 TB) /Month	$0.08 per GB	$0.15 per GB	$0.008 per GB	94% OFF	$0.0072 per GB
Next 100 TB (50 – 150 TB) /Month	$0.06 per GB	$0.13 per GB	$0.008 per GB	93% OFF	$0.0072 per GB
Next 350 TB (150 – 500 TB) /Month	$0.04 per GB	$0.11 per GB	$0.008 per GB	92% OFF	$0.0072 per GB
Next 524 TB (500 – 1,024 TB) /Month	$0.03 per GB	$0.10 per GB	$0.008 per GB	92% OFF	$0.0072 per GB
Next 4,096 TB (1,024 – 5,120 TB) /Month	$0.025 per GB	$0.09 per GB	$0.008 per GB	91% OFF	$0.0072 per GB

C、The Payment of BlockCDN Trade Platform.Based on ETH, BlockCDN will issue the token BCDN, which will be circulated in the trade platform to solve the trade payment problems of global sharers and the people who need acceleration.
In addition, the circulated amount of token BCDN will neither increase or decrease.The limited token provides service for constantly rapid growth of the massive CDN trading market.

3. How to invest in BlockCDN ？

1、**Join in BlockCDN to become sharing node：**

Share your idle internet devices to make them become CDN caching nodes and provide uploading traffic when the nearby clients need help to access websites. This can not only make you to gain benefits, but also make good utilization of resources and help to relieve the network congestion. When all of us join in sharing and become Nodes, it will also improve your access experience.

2、Join in BlockCDN to purchase your CDN

Join in BlockCDN. Go to the transparent, open and fair self-service trading platform to purchase your CDN service which best fits you. You can not only get CDN service with more nodes, higher speed, safer data and much more competitive price, but also support the sharing and utilization of energy and foster the growth of nodes. Therefore, your website will be accessed with a higher speed and improved access experience ,also easier to get CDN service at the best price.

3、Join in crowdfunding of BlockCDN

BlockCDN is the latest sharing trade platform and business model.Due to its joining in; it will bring a revolution to the CDN industry, just like UBER brought a revolution to taxi industry, which may cause hundreds of thousands of CDN practitioners to lose their jobs. However, this is a necessity of technological development.

Of course, if you are positive about the future development of BlockCDN, you can choose to be become a caching node or participant in BlockCDN's BCDN token crowdfunding in the near future.

According to the growth of CDN market and the prospect of the constant number of BCDN, if BCDN has a circulation of $12billion in the market, what the price of BCDN will be? What kind of investment product it will be? Therefore, there is no need to explain any more.

The following picture indexes the movements of prices of ETH in Poloniex:

4. What are the provisions and risks of investment in BlockCDN?

The crowdfunding provisions of BlockCDN ：

A、The final circulation of BCDN is based on the total confirmed quantity and 50% of total crowdfunding amount, among which 50% of crowdfunding amount of BCDN will be used as a reward for participants who join in sharing nodes within 3 years of launch of program.BCDN is issued by the means of mining. The mining is issued according to the number of uploading traffic on the node. The numbers of BCDN are neither increased or decreased.It is the constant number that joins in the circulation of CDN trade.

B、The compiling of 'Token' contract.The contract is compiled with Solidity(Reitwiessner and Wood [2015]).Each contract includes variables and Functions. So the outsiders can access the website by sending the means of transaction to Ethernet network,with the contract address of BCDN being the Receptor and method ID (parameter is optional) being the data. The ETH is directly sent to the specified address to finish crowdfunding.

C、According to time variable, the contract does not set up the crowfunding amount of BCDN but only set the time of total crowfunding as 28 days. The BCDN numbers got from crowfunding can be changeable. According to rules, in order to encourage the earlier Investor.The specific rules are as follows:
The first week and The second week:1ETH=100BCDN,
The first two days:1ETH=100*95%BCDN,
The second two days: 1ETH=100*95%*95%BCDN,
The third two days: 1ETH=100*95%*95%*95%BCDN,
The fourth two days: 1ETH=100*95%*95%*95%*95%BCDN,
The fifth two days: 1ETH=100*95%*95%*95%*95%*95%BCDN,
The the sixth two days: 1ETH=100*95%*95%*95%*95%*95%*95%BCDN,
The seventh two days: 1ETH=100*95%*95%*95%*95%*95%*95%*95%BCDN.

D、incentives for enrolling crowdfunding members.Pepole who recommends members to crowdfunding can get 5% of the total crowdfunding of BCDN as a reward that will be sent to the wallet of Refence people after crowdfunding completed.

E、All the BCDN holders can be rewarded with 40% of the total revenue of BlockCDN that will be sent to the wallet of holders with average gross. But the minimum reward will be greater or equal to 1ETH.The minimum reward that is less than 1ETH will be extended to next time in accumulation. The remaining 60% is used as expendure for CDN technology innovation, team maintainance, platform maintainance and undating.

F、Cooling-off period of investment is set during 28days of crowdfuding. Any investors who

want to withdraw can send BCDN to specified wallet. BlockCDN will refund investment money according to 1ETH=100BCDN.

G、If the aggregate amount of crowdfuding is less than 150,000ETH, it will be considered as a failure. All received ETH will be returned to investors, the token will be obsoleted, and the whole promotion incentives of members will be cancelled.

The Investment Risks of BlockCDN：

The Creation of BlockCDN tokens carries with it significant risk. Prior to Creating BlockCDN tokens, carefully consider the exemplary and non-exhaustive list of risks set forth below and, to the extent necessary, consult a lawyer, accountant, and/or tax professionals prior to Creating BlockCDN tokens.

1. Risk of Security Weaknesses in The BlockCDN's Software

The BlockCDN concept is both experimental in nature and unproven. There is a risk that, as an open source project, any contributor to The BlockCDN's software could introduce weaknesses or bugs into the BlockCDN software, causing the loss of BlockCDN tokens or ETH in one or more or even all of the BlockCDN Token Holder's accounts.

2. Risk of Weakness in the BlockCDN underlying blockchain, and/or Ethereum Network

The BlockCDN software is itself based on an unproven platform: the Ethereum blockchain. There is a risk that, as an open source project, any contributor to the Ethereum blockchain could introduce weaknesses or bugs into the Ethereum software, causing the loss of BlockCDN tokens or ETH in one or more or even all of the BlockCDN Token Holder's accounts.

3. Risk of unforeseen attack vectors

The field of Digital Cryptography is very new and for this reason, there is a risk of unforeseen attack both in terms of the underlying cryptographic protocol that back the functioning of the BlockCDN as well as 'game theory' related vectors which have not been documented to date. Both these vectors represent a risk that could lead the loss of BlockCDN tokens or ETH in one or more or even all of the BlockCDN Token Holder's accounts.

4. Regulatory risks

Blockchain technology and Ethereum are allowing new forms of interactions between individuals and/or companies, some of them are still to be imagined and implemented. Like with the appearance of cryptocurrencies such as Bitcoin, it is very likely that specific regulations will be set in different jurisdictions targeting blockchain technology and more specifically BlockCDNs. These regulations may or may not be BlockCDN friendly and some might even forbid any relationships between an individual or company and a BlockCDN.

Taxation

No party involved with the Creation of The BlockCDN makes any representations concerning the tax implications of the Creation of BlockCDN tokens or the possession or use of BlockCDN tokens. You bear the sole responsibility to determine if the Creation of BlockCDN tokens or the potential appreciation or depreciation in the value of BlockCDN tokens over time has tax implications for you in your home jurisdiction.
You Create BlockCDN tokens with your own actions. To the extent permitted by law, Third Parties or Individuals associated with the Creation of The BlockCDN are not liable for any tax liability associated with or arising from the Creation of BlockCDN tokens.

Forward looking statements

This document contains statements that are, or may be deemed to be, “forward looking statements” which are prospective in nature. These forward looking statements may be identified by the use of forward looking terminology, or the negative thereof such as “outlook”, "plans", "expects" or "does not expect", "is expected", "continues", "assumes", "is subject to", "budget", "scheduled", "estimates", "aims", "forecasts", "risks", "intends", "positioned", "predicts", "anticipates" or "does not anticipate", or "believes", or variations of such words or comparable terminology and phrases or statements that certain actions, events or results "may", "could", "should", “shall”, "would", "might" or "will" be taken, occur or be achieved. Such statements are qualified in their entirety by the inherent risks and uncertainties surrounding future expectations. Forward-looking statements are not based on historical facts, but rather on current predictions, expectations, beliefs, opinions, plans, objectives, goals, intentions and projections about future events, results of operations, prospects, financial condition and discussions of strategy. By their nature, forward looking statements involve known and unknown risks and uncertainties, many of which are beyond anyone's control.

Forward looking statements are not guarantees of future performance and may and often do differ materially from actual results. Important factors that could cause these uncertainties include, but are not limited to, those discussed in the “Risks and uncertainties” of this document. None of the third parties provide any representation, assurance or guarantee that the occurrence of the events expressed or implied in any forward-looking statements in this document will actually occur. You are cautioned not to place undue reliance on these forward-looking statements which only speak as of the date of this document. All third parties involved with the BlockCDN Creation and their affiliates expressly disclaim any intention, obligation or undertaking to update or revise any forward looking statements, whether as a result of new information, future events or otherwise. The making of this document does not constitute a recommendation regarding any securities.

Disclaimer of Warranties

THE USER EXPRESSLY AGREES THAT THE USER IS CREATING BlockCDN TOKENS AT THE USER’S SOLE RISK AND THAT BlockCDN TOKENS ARE CREATED ON

AN "AS IS" BASIS WITHOUT WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF TITLE OR IMPLIED WARRANTIES, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE (EXCEPT ONLY TO THE EXTENT PROHIBITED UNDER APPLICABLE LAW WITH ANY LEGALLY REQUIRED WARRANTY PERIOD TO THE SHORTER OF THIRTY DAYS FROM FIRST USE OR THE MINIMUM PERIOD REQUIRED).
WITHOUT LIMITING THE FOREGOING, NONE OF THIRD PARTIES OR INDIVIDUALS ASSOCIATED WITH THE BlockCDN CREATION WARRANT THAT THE PROCESS FOR PURCHASING BlockCDN TOKENS WILL BE UNINTERRUPTED OR ERROR-FREE.

Limitations Waiver of Liability

YOU ACKNOWLEDGE AND AGREE THAT, TO THE FULLEST EXTENT PERMITTED BY ANY APPLICABLE LAW, YOU WILL NOT HOLD THIRD PARTIES OR INDIVIDUALS ASSOCIATED WITH THE BlockCDN CREATION LIABLE FOR ANY AND ALL DAMAGES OR INJURY WHATSOEVER CAUSED BY OR RELATED TO USE OF, OR INABILITY TO USE, BlockCDN TOKENS OR THE BlockCDN PLATFORM UNDER ANY CAUSE OR ACTION WHATSOEVER OF ANY KIND IN ANY JURISDICTION, INCLUDING, WITHOUT LIMITATION, ACTIONS FOR BREACH OF WARRANTY, BREACH OF CONTRACT OR TORT (INCLUDING NEGLIGENCE) AND THAT NONE OF THE THIRD PARTIES OR INDIVIDUALS ASSOCIATED WITH THE BlockCDN CREATION SHALL BE LIABLE FOR ANY INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES, INCLUDING FOR LOSS OF PROFITS, GOODWILL OR DATA, IN ANY WAY WHATSOEVER ARISING OUT OF THE USE OF, OR INABILITY TO USE, OR CREATION OF, OR INABILITY TO CREATE, BlockCDN TOKENS.

YOU FURTHER SPECIFICALLY ACKNOWLEDGE THAT THE THIRD PARTIES OR INDIVIDUALS ASSOCIATED WITH THE BlockCDN CREATION ARE NOT LIABLE, AND YOU AGREE NOT TO SEEK TO HOLD ANY OF THE THIRD PARTIES OR INDIVIDUALS ASSOCIATED WITH THE BlockCDN CREATION LIABLE, FOR THE CONDUCT OF THIRD PARTIES, INCLUDING OTHER CREATORS OF BlockCDN TOKENS, AND THAT THE RISK OF CREATING AND USING BlockCDN TOKENS RESTS ENTIRELY WITH YOU.

5. How to achieve BlockCDN technically ？

How to make the idle equipment became CDN caching node

BlockCDN is intelligent deploying software of CDN node based on open source Squid and incorporates SDK and P2P technology. It gives full play to P2P and intelligent node scheduling of traditional CDN and SDK to realize multiple advantages such as, transformation from the serial data to parallel data, from single path to mufti-path, continuously optimize the algorithm and protocol, 100% of anti-stealing-link and anti-hijack. It can update intelligent device into CDN node and provide faster and better service efficiently.

Squid open source is as follows：https://github.com/squid-cache/squid

```
## Squid software is distributed under GPLv2+ license and includes
## contributions from numerous individuals and organizations.
## Please see the COPYING and CONTRIBUTORS files for details.
##

##
## AX_CXX_TYPE_NULLPTR shamelessly copied from the DUNE sources under GPL version 2
##
AC_DEFUN([AX_CXX_TYPE_NULLPTR],[
  AC_REQUIRE([AC_PROG_CXX])
  AC_LANG_PUSH([C++])
  AC_MSG_CHECKING([whether nullptr is supported])
  AC_TRY_COMPILE([],[char* ch = nullptr;], [
    HAVE_NULLPTR=yes
    AC_MSG_RESULT(yes)], [
    HAVE_NULLPTR=no
    AC_MSG_RESULT(no)])
  if test "x$HAVE_NULLPTR" = xno; then
    AC_DEFINE(nullptr, NULL, [Leave undefined if nullptr is supported])
  fi
  AC_MSG_CHECKING([whether nullptr_t is supported])
  AC_TRY_COMPILE([#include <cstddef>],[typedef nullptr_t peng;], [
    HAVE_NULLPTR_T=yes
    AC_MSG_RESULT(yes)], [
    HAVE_NULLPTR_T=no
    AC_MSG_RESULT(no)])
```

```
  if test "x$HAVE_NULLPTR_T" = xyes; then
    AC_DEFINE(HAVE_NULLPTR_T, 1, [Define to 1 if nullptr_t is supported])
  fi
  AC_LANG_POP
])

## Hand crafted for Squid under GPL version 2
AC_DEFUN([AX_CXX_TYPE_UNIQUE_PTR],[
  AC_REQUIRE([AC_PROG_CXX])
  AC_LANG_PUSH([C++])
  AC_MSG_CHECKING([whether std::unique_ptr<T> is supported])
  AC_TRY_COMPILE([#include <memory>],[std::unique_ptr<char> c;], [
    HAVE_UNIQUE_PTR=yes
    AC_MSG_RESULT(yes)], [
    HAVE_UNIQUE_PTR=no
    AC_MSG_RESULT(no)])
  if test "x$HAVE_UNIQUE_PTR" = xno; then
    AC_DEFINE(unique_ptr, auto_ptr, [Leave undefined if std::unique_ptr<T> is supported])
  fi
  if test "x$HAVE_UNIQUE_PTR" = xyes; then
    AC_DEFINE(HAVE_UNIQUE_PTR, 1, [Define to 1 if std::unique_ptr<T> is supported])
  fi
  AC_LANG_POP
])

## Hand crafted for Squid under GPL version 2
AC_DEFUN([AX_CXX_TYPE_UNIFORM_DISTRIBUTIONS],[
  AC_REQUIRE([AC_PROG_CXX])
  AC_LANG_PUSH([C++])
  AC_CHECK_HEADERS(tr1/random)
  AC_CACHE_CHECK([whether std::uniform_int_distribution<T> is supported],
                 [squid_cv_std_uniform_int_distribution_works],[
    AC_TRY_COMPILE([#include <random>],[std::uniform_int_distribution<int> c;],
      [squid_cv_std_uniform_int_distribution_works=yes],
      [squid_cv_std_uniform_int_distribution_works=no])
    ])
  SQUID_DEFINE_BOOL([HAVE_STD_UNIFORM_INT_DISTRIBUTION],
      [$squid_cv_std_uniform_int_distribution_works],
      [Define if c++11 std::uniform_int_distribution is supported])

  AC_CACHE_CHECK([whether std::uniform_real_distribution<T> is supported],
                 [squid_cv_std_uniform_real_distribution_works],[
    AC_REQUIRE([AC_PROG_CXX])
    AC_LANG_PUSH([C++])
```

```
      AC_TRY_COMPILE([#include <random>],[std::uniform_real_distribution<double> c;],
        [squid_cv_std_uniform_real_distribution_works=yes],
        [squid_cv_std_uniform_real_distribution_works=no])
      ])
   SQUID_DEFINE_BOOL([HAVE_STD_UNIFORM_REAL_DISTRIBUTION],
        [$squid_cv_std_uniform_real_distribution_works],
        [Define if c++11 std::uniform_real_distribution is supported])

   AC_LANG_POP
])

## SQUID_CXX_STD_UNDERLYING_TYPE
## checks whether the std::underlying_type<enumType>::type trait exists
AC_DEFUN([SQUID_CXX_STD_UNDERLYING_TYPE],[
   AC_CACHE_CHECK([whether compiler supports std::underlying_type],
      [squid_cv_have_std_underlying_type],[
         AC_REQUIRE([AC_PROG_CXX])
         AC_LANG_PUSH([C++])
         AC_COMPILE_IFELSE([
            AC_LANG_PROGRAM([
#include <type_traits>
enum class testEnum { one, two, three };
            ],[
            std::underlying_type<testEnum>::type testNum = 0;
            ])],
            [squid_cv_have_std_underlying_type=yes],
            [squid_cv_have_std_underlying_type=no])
         AC_LANG_POP
   ])
   SQUID_DEFINE_BOOL([HAVE_STD_UNDERLYING_TYPE],
       [$squid_cv_have_std_underlying_type],
       [Define if stdlibc support std::underlying_type for enums])
])

#serial 4

m4_define([_AX_CXX_COMPILE_STDCXX_11_testbody], [[
      template <typename T>
      struct check
      {
         static_assert(sizeof(int) <= sizeof(T), "not big enough"); // GCC 4.3+
      };

#if WHEN_SQUID_HAS_MANDATORY_GCC_4_789_SUPPORT
```

```
      struct Base {
      virtual void f() {}
      };
      struct Child : public Base {
      virtual void f() override {} // GCC 4.7+
      };
#endif

      typedef check<check<bool>> right_angle_brackets; // GCC 4.3+

      int a;
      decltype(a) b; // GCC 4.3+

      typedef check<int> check_type;
      check_type c;
      check_type&& cr = static_cast<check_type&&>(c); // GCC 4.3+

      auto d = a;          // GCC 4.4+
#if WHEN_SQUID_HAS_MANDATORY_GCC_4_789_SUPPORT
      auto l = [](){}; // GCC 4.5+ (void lambda seems not to be documented)
#endif
]])

AC_DEFUN([AX_CXX_COMPILE_STDCXX_11], [dnl
  m4_if([$1], [], [],
        [$1], [ext], [],
        [$1], [noext], [],
        [m4_fatal([invalid argument `$1' to AX_CXX_COMPILE_STDCXX_11])])dnl
  m4_if([$2], [], [ax_cxx_compile_cxx11_required=true],
        [$2], [mandatory], [ax_cxx_compile_cxx11_required=true],
        [$2], [optional], [ax_cxx_compile_cxx11_required=false],
        [m4_fatal([invalid second argument `$2' to AX_CXX_COMPILE_STDCXX_11])])
  AC_LANG_PUSH([C++])dnl
  ac_success=no
  AC_CACHE_CHECK(whether $CXX supports C++11 features by default,
  ax_cv_cxx_compile_cxx11,
  [AC_COMPILE_IFELSE([AC_LANG_SOURCE([_AX_CXX_COMPILE_STDCXX_11_testbody])],
    [ax_cv_cxx_compile_cxx11=yes],
    [ax_cv_cxx_compile_cxx11=no])])
  if test x$ax_cv_cxx_compile_cxx11 = xyes; then
    ac_success=yes
  fi

  m4_if([$1], [noext], [], [dnl
```

```
  if test x$ac_success = xno; then
    for switch in -std=gnu++11 -std=gnu++0x; do
      cachevar=AS_TR_SH([ax_cv_cxx_compile_cxx11_$switch])
      AC_CACHE_CHECK(whether $CXX supports C++11 features with $switch,
                     $cachevar,
        [ac_save_CXXFLAGS="$CXXFLAGS"
         CXXFLAGS="$CXXFLAGS $switch"

AC_COMPILE_IFELSE([AC_LANG_SOURCE([_AX_CXX_COMPILE_STDCXX_11_testbody])],
          [eval $cachevar=yes],
          [eval $cachevar=no])
         CXXFLAGS="$ac_save_CXXFLAGS"])
      if eval test x\$$cachevar = xyes; then
        CXXFLAGS="$CXXFLAGS $switch"
        ac_success=yes
        break
      fi
    done
  fi])

  m4_if([$1], [ext], [], [dnl
  if test x$ac_success = xno; then
    for switch in -std=c++11 -std=c++0x; do
      cachevar=AS_TR_SH([ax_cv_cxx_compile_cxx11_$switch])
      AC_CACHE_CHECK(whether $CXX supports C++11 features with $switch,
                     $cachevar,
        [ac_save_CXXFLAGS="$CXXFLAGS"
         CXXFLAGS="$CXXFLAGS $switch"

AC_COMPILE_IFELSE([AC_LANG_SOURCE([_AX_CXX_COMPILE_STDCXX_11_testbody])],
          [eval $cachevar=yes],
          [eval $cachevar=no])
         CXXFLAGS="$ac_save_CXXFLAGS"])
      if eval test x\$$cachevar = xyes; then
        CXXFLAGS="$CXXFLAGS $switch"
        ac_success=yes
        break
      fi
    done
  fi])
  AC_LANG_POP([C++])
  if test x$ax_cxx_compile_cxx11_required = xtrue; then
    if test x$ac_success = xno; then
      AC_MSG_ERROR([*** A compiler with support for C++11 language features is required.])
```

```
    fi
  else
    if test x$ac_success = xno; then
      HAVE_CXX11=0
      AC_MSG_NOTICE([No compiler with C++11 support was found])
    else
      HAVE_CXX11=1
      AC_DEFINE(HAVE_CXX11,1,
                [define if the compiler supports basic C++11 syntax])
    fi

    AC_SUBST(HAVE_CXX11)
  fi
])

## Squid software is distributed under GPLv2+ license and includes
## contributions from numerous individuals and organizations.
## Please see the COPYING and CONTRIBUTORS files for details.
##

dnl =========================================================================
dnl                  http://autoconf-archive.cryp.to/ax_with_prog.html
dnl =========================================================================
dnl
dnl SYNOPSIS
dnl
dnl   AX_WITH_PROG([VARIABLE],[program],[VALUE-IF-NOT-FOUND],[PATH])
dnl
dnl DESCRIPTION
dnl
dnl   Locates an installed program binary, placing the result in the precious
dnl   variable VARIABLE. Accepts a present VARIABLE, then --with-program, and
dnl   failing that searches for program in the given path (which defaults to
dnl   the system path). If program is found, VARIABLE is set to the full path
dnl   of the binary; if it is not found VARIABLE is set to VALUE-IF-NOT-FOUND
dnl   if provided, unchanged otherwise.
dnl
dnl   A typical example could be the following one:
dnl
dnl           AX_WITH_PROG(PERL,perl)
dnl
dnl   NOTE: This macro is based upon the original AX_WITH_PYTHON macro from
dnl   Dustin J. Mitchell <dustin@cs.uchicago.edu>.
dnl
```

```
dnl LAST MODIFICATION
dnl
dnl     2008-05-05
dnl
dnl COPYLEFT
dnl
dnl     Copyright (c) 2008 Francesco Salvestrini <salvestrini@users.sourceforge.net>
dnl     Copyright (c) 2008 Dustin J. Mitchell <dustin@cs.uchicago.edu>
dnl
dnl     Copying and distribution of this file, with or without modification, are
dnl     permitted in any medium without royalty provided the copyright notice
dnl     and this notice are preserved.
dnl
AC_DEFUN([AX_WITH_PROG],[
      AC_PREREQ([2.61])

      pushdef([VARIABLE],$1)
      pushdef([EXECUTABLE],$2)
      pushdef([VALUE_IF_NOT_FOUND],$3)
      pushdef([PATH_PROG],$4)

      AC_ARG_VAR(VARIABLE,Absolute path to EXECUTABLE executable)

      AS_IF(test -z "$VARIABLE",[
            AC_MSG_CHECKING(whether EXECUTABLE executable path has been provided)

AC_ARG_WITH(EXECUTABLE,AS_HELP_STRING([--with-EXECUTABLE=[[[[PATH]]]]],absolute path to
EXECUTABLE executable), [
            AS_IF([test "$withval" != "yes"],[
                  VARIABLE="$withval"
            AC_MSG_RESULT($VARIABLE)
            ],[
            VARIABLE=""
                  AC_MSG_RESULT([no])
            ])
      ],[
            AC_MSG_RESULT([no])
      ])

            AS_IF(test -z "$VARIABLE",[

AC_PATH_PROG([]VARIABLE[],[]EXECUTABLE[],[]VALUE_IF_NOT_FOUND[],[]PATH_PROG[])
            ])
      ])
```

```
    popdef([PATH_PROG])
    popdef([VALUE_IF_NOT_FOUND])
    popdef([EXECUTABLE])
    popdef([VARIABLE])
])

AC_DEFUN([SQUID_CC_CHECK_ARGUMENT],[
  AC_CACHE_CHECK([whether compiler accepts $2],[$1],
  [{
    AC_REQUIRE([AC_PROG_CC])
    SAVED_FLAGS="$CFLAGS"
    SAVED_CXXFLAGS="$CXXFLAGS"
    CFLAGS="$CFLAGS $2"
    CXXFLAGS="$CXXFLAGS $2"
    AC_TRY_LINK([],[int foo; ],
      [$1=yes],[$1=no])
    CFLAGS="$SAVED_CFLAGS"
    CXXFLAGS="$SAVED_CXXFLAGS"
  }])
])

# Check if the compiler requires a supplied flag to build a test program.
# When cross-compiling set flags explicitly.
#
# first argument is the variable containing the result
# (will be set to "yes" or "no")
# second argument is the flag to be tested, verbatim
# third is the #include and global setup for test program, verbatim
# fourth is the test program to compile, verbatim
#
AC_DEFUN([SQUID_CC_REQUIRE_ARGUMENT],[
  AC_CACHE_CHECK([whether compiler requires $2],[$1],
  [{
    AC_REQUIRE([AC_PROG_CC])
    SAVED_FLAGS="$CFLAGS"
    SAVED_CXXFLAGS="$CXXFLAGS"
    AC_COMPILE_IFELSE([AC_LANG_PROGRAM($3,$4)],[$1=no],[],[$1=no])
    if test "x$1" != "xno" ; then
      CFLAGS="$CFLAGS $2"
      CXXFLAGS="$CXXFLAGS $2"
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM($3,$4)],[$1=yes],[$1=no],[$1=no])
    fi
    CFLAGS="$SAVED_CFLAGS"
```

```
      CXXFLAGS="$SAVED_CXXFLAGS"
    }])
])

# detect what kind of compiler we're using, either by using hints from
# autoconf itself, or by using predefined preprocessor macros
# sets the variable squid_cv_compiler to one of
#    - gcc
#    - sunstudio
#    - none (undetected)
#
AC_DEFUN([SQUID_CC_GUESS_VARIANT], [
  AC_CACHE_CHECK([what kind of compiler we're using],[squid_cv_compiler],
  [
    AC_REQUIRE([AC_PROG_CC])
    dnl repeat the next block for each compiler, changing the
    dnl preprocessor definition so that it depends on platform-specific
    dnl predefined macros
    dnl SunPro CC
    if test -z "$squid_cv_compiler" ; then
      AC_COMPILE_IFELSE([
        AC_LANG_PROGRAM([[
#if !defined(__SUNPRO_C) && !defined(__SUNPRO_CC)
#error "not sunpro c"
#endif
        ]])],[squid_cv_compiler="sunstudio"],[])
    fi
    dnl Intel CC
    if test -z "$squid_cv_compiler" ; then
      AC_COMPILE_IFELSE([
        AC_LANG_PROGRAM([[
#if !defined(__ICC)
#error "not Intel(R) C++ Compiler"
#endif
        ]])],[squid_cv_compiler="icc"],[])
    fi
    dnl clang
    if test -z "$squid_cv_compiler" ; then
      AC_COMPILE_IFELSE([
        AC_LANG_PROGRAM([[
#if !defined(__clang__)
#error "not clang"
#endif
        ]])],[squid_cv_compiler="clang"],[])
```

```
    fi
    dnl microsoft visual c++
    if test -z "$squid_cv_compiler" ; then
      AC_COMPILE_IFELSE([
        AC_LANG_PROGRAM([[
#if !defined(_MSC_VER)
#error "not Microsoft VC++"
#endif
        ]])],[squid_cv_compiler="msvc"],[])
    fi
    dnl gcc. MUST BE LAST as many other compilers also define it for compatibility
    if test -z "$squid_cv_compiler" ; then
      AC_COMPILE_IFELSE([
        AC_LANG_PROGRAM([[
#if !defined(__GNUC__)
#error "not gcc"
#endif
        ]])],[squid_cv_compiler="gcc"],[])
    fi
    dnl end of block to be repeated
    if test -z "$squid_cv_compiler" ; then
      squid_cv_compiler="none"
    fi
    ]) dnl AC_CACHE_CHECK
  ]) dnl AC_DEFUN

# define the flag to use to have the compiler treat warnings as errors
# requirs SQUID_CC_GUESS_VARIANT
# Sets a few variables to contain some compiler-dependent command line
# options, or to empty strings if the compiler doesn't support those
# options
# They are (with their GCC equivalent):
# squid_cv_cc_option_werror      (-Werror)
# squid_cv_cxx_option_werror     (-Werror)
# squid_cv_cc_option_wall        (-Wall)
# squid_cv_cc_option_optimize (-O3)
#
AC_DEFUN([SQUID_CC_GUESS_OPTIONS], [
  AC_REQUIRE([SQUID_CC_GUESS_VARIANT])
  AC_MSG_CHECKING([for compiler variant])
  case "$squid_cv_compiler" in
    gcc)
      squid_cv_cc_option_werror="-Werror"
      squid_cv_cxx_option_werror="-Werror"
```

```
    squid_cv_cc_option_wall="-Wall"
    squid_cv_cc_option_optimize="-O3"
    squid_cv_cc_arg_pipe="-pipe"
    ;;
  sunstudio)
    squid_cv_cc_option_werror="-errwarn=%all -errtags"
    squid_cv_cxx_option_werror="-errwarn=%all,no%badargtype2w,no%wbadinit,no%wbadasg -errtags"
    squid_cv_cc_option_wall="+w"
    squid_cv_cc_option_optimize="-fast"
    squid_cv_cc_arg_pipe=""
    ;;
  clang)
    squid_cv_cxx_option_werror="-Werror -Qunused-arguments"
    squid_cv_cc_option_werror="$squid_cv_cxx_option_werror"
    squid_cv_cc_option_wall="-Wall"
    squid_cv_cc_option_optimize="-O2"
    squid_cv_cc_arg_pipe=""
    ;;
  icc)
    squid_cv_cxx_option_werror="-Werror"
    squid_cv_cc_option_werror="$squid_cv_cxx_option_werror"
    squid_cv_cc_option_wall="-Wall"
    squid_cv_cc_option_optimize="-O2"
    squid_cv_cc_arg_pipe=""
    ;;
  *)
    squid_cv_cxx_option_werror=""
    squid_cv_cc_option_werror=""
    squid_cv_cc_option_wall=""
    squid_cv_cc_option_optimize="-O"
    squid_cv_cc_arg_pipe=""
    ;;
 esac
 AC_MSG_RESULT([$squid_cv_compiler])
])

dnl This encapsulates the nasty mess of headers we need to check when
dnl checking types.
AC_DEFUN([SQUID_DEFAULT_INCLUDES],[[
/* What a mess.. many systems have added the (now standard) bit types
 * in their own ways, so we need to scan a wide variety of headers to
 * find them..
 * IMPORTANT: Keep compat/types.h syncronised with this list
 */
```

```
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_LINUX_TYPES_H
#include <linux/types.h>
#endif
#if HAVE_STDLIB_H
#include <stdlib.h>
#endif
#if HAVE_STDDEF_H
#include <stddef.h>
#endif
#if HAVE_INTTYPES_H
#include <inttypes.h>
#endif
#if HAVE_SYS_BITYPES_H
#include <sys/bitypes.h>
#endif
#if HAVE_SYS_SELECT_H
#include <sys/select.h>
#endif
#if HAVE_NETINET_IN_SYSTM_H
#include <netinet/in_systm.h>
#endif
]])

dnl *BSD net headers
AC_DEFUN([SQUID_BSDNET_INCLUDES],[
SQUID_DEFAULT_INCLUDES
#if HAVE_SYS_PARAM_H
#include <sys/param.h>
#endif
#if HAVE_SYS_TIME_H
#include <sys/time.h>
#endif
#if HAVE_SYS_SOCKET_H
#include <sys/socket.h>
#endif
#if HAVE_NET_IF_H
#include <net/if.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
```

```
#if HAVE_NETINET_IP_H
#include <netinet/ip.h>
#endif
#if HAVE_NETINET_IP_COMPAT_H
#include <netinet/ip_compat.h>
#endif
#if HAVE_NETINET_IP_FIL_H
#include <netinet/ip_fil.h>
#endif
])

dnl these checks must be performed in the same order as here defined,
dnl and have mostly been lifted out of an inlined configure.ac.

dnl checks for a broken solaris header file, and sets squid_cv_broken_krb5_h
dnl to yes if that's the case
AC_DEFUN([SQUID_CHECK_KRB5_SOLARIS_BROKEN_KRB5_H], [
   AC_CACHE_CHECK([for broken Solaris krb5.h],squid_cv_broken_krb5_h, [
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <krb5.h>
int i;
]])], [ squid_cv_broken_krb5_h=no ], [
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#if defined(__cplusplus)
#define KRB5INT_BEGIN_DECLS         extern "C" {
#define KRB5INT_END_DECLS
KRB5INT_BEGIN_DECLS
#endif
#include <krb5.h>
int i;
]])], [ squid_cv_broken_krb5_h=yes ], [ squid_cv_broken_krb5_h=no ])
      ])
   ])
]) dnl SQUID_CHECK_KRB5_SOLARIS_BROKEN_KRB5_H


AC_DEFUN([SQUID_CHECK_KRB5_HEIMDAL_BROKEN_KRB5_H], [
   AC_CACHE_CHECK([for broken Heimdal krb5.h],squid_cv_broken_heimdal_krb5_h, [
      AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <krb5.h>
int
main(void)
{
        krb5_context context;
```

```
        krb5_init_context(&context);

        return 0;
}
]])], [ squid_cv_broken_heimdal_krb5_h=no ], [
    AC_RUN_IFELSE([AC_LANG_SOURCE([[
#if defined(__cplusplus)
extern "C" {
#endif
#include <krb5.h>
#if defined(__cplusplus)
}
#endif
int
main(void)
{
        krb5_context context;

        krb5_init_context(&context);

        return 0;
}
]])], [ squid_cv_broken_heimdal_krb5_h=yes ], [ squid_cv_broken_heimdal_krb5_h=no ])
    ])
  ])
]) dnl SQUID_CHECK_KRB5_HEIMDAL_BROKEN_KRB5_H

dnl check the max skew in the krb5 context, and sets squid_cv_max_skew_context
AC_DEFUN([SQUID_CHECK_MAX_SKEW_IN_KRB5_CONTEXT],[
  AC_CACHE_CHECK([for max_skew in struct krb5_context],
                 squid_cv_max_skew_context, [
    AC_COMPILE_IFELSE([
      AC_LANG_PROGRAM([[
#if HAVE_BROKEN_SOLARIS_KRB5_H
#if defined(__cplusplus)
#define KRB5INT_BEGIN_DECLS     extern "C" {
#define KRB5INT_END_DECLS
KRB5INT_BEGIN_DECLS
#endif
#endif
#if USE_APPLE_KRB5
#define KERBEROS_APPLE_DEPRECATED(x)
#endif
```

```
#include <krb5.h>
krb5_context kc; kc->max_skew = 1;
          ]])
      ],[ squid_cv_max_skew_context=yes ],
      [ squid_cv_max_skew_context=no ])
   ])
])

dnl check whether the kerberos context has a memory cache. Sets
dnl squid_cv_memory_cache if that's the case.
AC_DEFUN([SQUID_CHECK_KRB5_CONTEXT_MEMORY_CACHE],[
   AC_CACHE_CHECK([for memory cache], squid_cv_memory_cache, [
      AC_RUN_IFELSE([
         AC_LANG_SOURCE([[
#if HAVE_BROKEN_SOLARIS_KRB5_H
#if defined(__cplusplus)
#define KRB5INT_BEGIN_DECLS         extern "C" {
#define KRB5INT_END_DECLS
KRB5INT_BEGIN_DECLS
#endif
#endif
#if USE_APPLE_KRB5
#define KERBEROS_APPLE_DEPRECATED(x)
#endif
#include <krb5.h>
int main(int argc, char *argv[])
{
      krb5_context context;
      krb5_ccache cc;

      krb5_init_context(&context);
      return krb5_cc_resolve(context, "MEMORY:test_cache", &cc);
}
          ]])
      ], [ squid_cv_memory_cache=yes ], [ squid_cv_memory_cache=no ], [:])
   ])
])

dnl check whether the kerberos context has a memory keytab. Sets
dnl squid_cv_memory_keytab if that's the case.
AC_DEFUN([SQUID_CHECK_KRB5_CONTEXT_MEMORY_KEYTAB],[
   AC_CACHE_CHECK([for memory keytab], squid_cv_memory_keytab, [
      AC_RUN_IFELSE([
         AC_LANG_SOURCE([[
```

```
#if HAVE_BROKEN_SOLARIS_KRB5_H
#if defined(__cplusplus)
#define KRB5INT_BEGIN_DECLS         extern "C" {
#define KRB5INT_END_DECLS
KRB5INT_BEGIN_DECLS
#endif
#endif
#if USE_APPLE_KRB5
#define KERBEROS_APPLE_DEPRECATED(x)
#endif
#include <krb5.h>
int main(int argc, char *argv[])
{
       krb5_context context;
       krb5_keytab kt;

       krb5_init_context(&context);
       return krb5_kt_resolve(context, "MEMORY:test_keytab", &kt);
}
          ]])
       ], [ squid_cv_memory_keytab=yes ], [ squid_cv_memory_keytab=no ], [:])
   ])
])


dnl checks that gssapi is ok, and sets squid_cv_working_gssapi accordingly
AC_DEFUN([SQUID_CHECK_WORKING_GSSAPI], [
   AC_CACHE_CHECK([for working gssapi], squid_cv_working_gssapi, [
      AC_RUN_IFELSE([AC_LANG_SOURCE([[
#if USE_HEIMDAL_KRB5
#if HAVE_GSSAPI_GSSAPI_H
#include <gssapi/gssapi.h>
#elif HAVE_GSSAPI_H
#include <gssapi.h>
#endif
#elif USE_GNUGSS
#if HAVE_GSS_H
#include <gss.h>
#endif
#else
#if USE_APPLE_KRB5
#define GSSKRB_APPLE_DEPRECATED(x)
#endif
#if HAVE_GSSAPI_GSSAPI_H
```

```
#include <gssapi/gssapi.h>
#elif HAVE_GSSAPI_H
#include <gssapi.h>
#endif
#if HAVE_GSSAPI_GSSAPI_KRB5_H
#include <gssapi/gssapi_krb5.h>
#endif
#if HAVE_GSSAPI_GSSAPI_GENERIC_H
#include <gssapi/gssapi_generic.h>
#endif
#endif
int
main(void)
{
        OM_uint32 val;
        gss_OID_set set;

        gss_create_empty_oid_set(&val, &set);

        return 0;
}
  ]])],    [ squid_cv_working_gssapi=yes ], [ squid_cv_working_gssapi=no ], [:])])
if test "x$squid_cv_working_gssapi" = "xno" -a `echo $LIBS | grep -i -c "\-L"` -gt 0; then
  AC_MSG_NOTICE([Check Runtime library path !])
fi
])

dnl check for a working spnego, and set squid_cv_have_spnego
AC_DEFUN([SQUID_CHECK_SPNEGO_SUPPORT], [
  AC_CACHE_CHECK([for spnego support], squid_cv_have_spnego, [
    AC_RUN_IFELSE([AC_LANG_SOURCE([[
#if USE_HEIMDAL_KRB5
#if HAVE_GSSAPI_GSSAPI_H
#include <gssapi/gssapi.h>
#elif HAVE_GSSAPI_H
#include <gssapi.h>
#endif
#elif USE_GNUGSS
#if HAVE_GSS_H
#include <gss.h>
#endif
#else
#if USE_APPLE_KRB5
#define GSSKRB_APPLE_DEPRECATED(x)
```

```
#endif
#if HAVE_GSSAPI_GSSAPI_H
#include <gssapi/gssapi.h>
#elif HAVE_GSSAPI_H
#include <gssapi.h>
#endif
#if HAVE_GSSAPI_GSSAPI_KRB5_H
#include <gssapi/gssapi_krb5.h>
#endif
#if HAVE_GSSAPI_GSSAPI_GENERIC_H
#include <gssapi/gssapi_generic.h>
#endif
#endif
#include <string.h>
int main(int argc, char *argv[]) {
  OM_uint32 major_status,minor_status;
  gss_OID_set gss_mech_set;
  int i;

static gss_OID_desc _gss_mech_spnego    = {6, (void *)"\x2b\x06\x01\x05\x05\x02"};
gss_OID gss_mech_spnego = &_gss_mech_spnego;

  major_status = gss_indicate_mechs( &minor_status, &gss_mech_set);

  for (i=0;i<gss_mech_set->count;i++) {
        if
(!memcmp(gss_mech_set->elements[i].elements,gss_mech_spnego->elements,gss_mech_set->element
s[i].length)) {
              return 0;
        }
  }

  return 1;
}
    ]])],
    [ squid_cv_have_spnego=yes ], [ squid_cv_have_spnego=no ],[:])])
])

dnl checks that krb5 is functional. Sets squid_cv_working_krb5
AC_DEFUN([SQUID_CHECK_WORKING_KRB5],[
    AC_CACHE_CHECK([for working krb5], squid_cv_working_krb5, [
       AC_RUN_IFELSE([AC_LANG_SOURCE([[
#if USE_APPLE_KRB5
#define KERBEROS_APPLE_DEPRECATED(x)
```

```
#endif
#if HAVE_KRB5_H
#if HAVE_BROKEN_SOLARIS_KRB5_H
#if defined(__cplusplus)
#define KRB5INT_BEGIN_DECLS         extern "C" {
#define KRB5INT_END_DECLS
KRB5INT_BEGIN_DECLS
#endif
#endif
#if HAVE_BROKEN_HEIMDAL_KRB5_H
extern "C" {
#include <krb5.h>
}
#else
#include <krb5.h>
#endif
#endif

int
main(void)
{
        krb5_context context;

        krb5_init_context(&context);

        return 0;
}
   ]])], [ squid_cv_working_krb5=yes ], [ squid_cv_working_krb5=no ],[:])])
if test "x$squid_cv_working_krb5" = "xno" -a `echo $LIBS | grep -i -c "\-L"` -gt 0; then
   AC_MSG_NOTICE([Check Runtime library path !])
fi
])


dnl checks for existence of krb5 functions
AC_DEFUN([SQUID_CHECK_KRB5_FUNCS],[

   ac_com_error_message=no
   if test "x$ac_cv_header_com_err_h" = "xyes" ; then
      AC_EGREP_HEADER(error_message,com_err.h,ac_com_error_message=yes)
   elif test "x$ac_cv_header_et_com_err_h" = "xyes" ; then
      AC_EGREP_HEADER(error_message,et/com_err.h,ac_com_error_message=yes)
   fi
```

```
if test `echo $KRB5LIBS | grep -c com_err` -ne 0 -a "x$ac_com_error_message" = "xyes" ; then
   AC_CHECK_LIB(com_err,error_message,
      AC_DEFINE(HAVE_ERROR_MESSAGE,1,
         [Define to 1 if you have error_message]),)
elif test    "x$ac_com_error_message" = "xyes" ; then
   AC_CHECK_LIB(krb5,error_message,
      AC_DEFINE(HAVE_ERROR_MESSAGE,1,
         [Define to 1 if you have error_message]),)
fi

AC_CHECK_LIB(krb5,krb5_get_err_text,
   AC_DEFINE(HAVE_KRB5_GET_ERR_TEXT,1,
      [Define to 1 if you have krb5_get_err_text]),)
AC_CHECK_LIB(krb5,krb5_get_error_message,
   AC_DEFINE(HAVE_KRB5_GET_ERROR_MESSAGE,1,
      [Define to 1 if you have krb5_get_error_message]),)
AC_CHECK_LIB(krb5,krb5_free_error_message,
   AC_DEFINE(HAVE_KRB5_FREE_ERROR_MESSAGE,1,
      [Define to 1 if you have krb5_free_error_message]),)
AC_CHECK_LIB(krb5,krb5_free_error_string,
   AC_DEFINE(HAVE_KRB5_FREE_ERROR_STRING,1,
      [Define to 1 if you have krb5_free_error_string]),)
AC_CHECK_DECLS(krb5_kt_free_entry,,,[#include <krb5.h>])
AC_CHECK_TYPE(krb5_pac,
   AC_DEFINE(HAVE_KRB5_PAC,1,
      [Define to 1 if you have krb5_pac]),,
      [#include <krb5.h>])
AC_CHECK_LIB(krb5,krb5_kt_free_entry,
   AC_DEFINE(HAVE_KRB5_KT_FREE_ENTRY,1,
      [Define to 1 if you have krb5_kt_free_entry]),)
AC_CHECK_LIB(krb5,krb5_get_init_creds_keytab,
   AC_DEFINE(HAVE_GET_INIT_CREDS_KEYTAB,1,
      [Define to 1 if you have krb5_get_init_creds_keytab]),)
AC_CHECK_LIB(krb5,krb5_get_max_time_skew,
   AC_DEFINE(HAVE_KRB5_GET_MAX_TIME_SKEW,1,
      [Define to 1 if you have krb5_get_max_time_skew]),)
AC_CHECK_LIB(krb5,krb5_get_profile,
   AC_DEFINE(HAVE_KRB5_GET_PROFILE,1,
      [Define to 1 if you have krb5_get_profile]),)
AC_CHECK_LIB(krb5,profile_get_integer,
   AC_DEFINE(HAVE_PROFILE_GET_INTEGER,1,
      [Define to 1 if you have profile_get_integer]),)
AC_CHECK_LIB(krb5,profile_release,
   AC_DEFINE(HAVE_PROFILE_RELEASE,1,
```

```
      [Define to 1 if you have profile_release]),)
AC_CHECK_LIB(krb5,krb5_get_renewed_creds,
   AC_DEFINE(HAVE_KRB5_GET_RENEWED_CREDS,1,
      [Define to 1 if you have krb5_get_renewed_creds]),)
AC_CHECK_LIB(krb5,krb5_principal_get_realm,
   AC_DEFINE(HAVE_KRB5_PRINCIPAL_GET_REALM,1,
      [Define to 1 if you have krb5_principal_get_realm]),)
AC_CHECK_LIB(krb5, krb5_get_init_creds_opt_alloc,
   AC_DEFINE(HAVE_KRB5_GET_INIT_CREDS_OPT_ALLOC,1,
      [Define to 1 if you have krb5_get_init_creds_opt_alloc]),)
AC_MSG_CHECKING([for krb5_get_init_creds_free requires krb5_context])
AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
         #if USE_APPLE_KRB5
         #define KERBEROS_APPLE_DEPRECATED(x)
         #endif
   #include <krb5.h>
   ]],[[krb5_context context;
    krb5_get_init_creds_opt *options;
    krb5_get_init_creds_opt_free(context, options)]])],[
   AC_DEFINE(HAVE_KRB5_GET_INIT_CREDS_FREE_CONTEXT,1,
            [Define to 1 if you krb5_get_init_creds_free requires krb5_context])
   AC_MSG_RESULT(yes)
   ],[AC_MSG_RESULT(no)],[AC_MSG_RESULT(no)])

AC_CHECK_FUNCS(gss_map_name_to_any,
   AC_DEFINE(HAVE_GSS_MAP_ANY_TO_ANY,1,
      [Define to 1 if you have gss_map_name_to_any]),)
AC_CHECK_FUNCS(gsskrb5_extract_authz_data_from_sec_context,
   AC_DEFINE(HAVE_GSSKRB5_EXTRACT_AUTHZ_DATA_FROM_SEC_CONTEXT,1,
      [Define to 1 if you have gsskrb5_extract_authz_data_from_sec_context]),)

SQUID_CHECK_KRB5_CONTEXT_MEMORY_CACHE
SQUID_DEFINE_BOOL(HAVE_KRB5_MEMORY_CACHE,$squid_cv_memory_cache,
       [Define if kerberos has MEMORY: cache support])

SQUID_CHECK_KRB5_CONTEXT_MEMORY_KEYTAB
SQUID_DEFINE_BOOL(HAVE_KRB5_MEMORY_KEYTAB,$squid_cv_memory_keytab,
       [Define if kerberos has MEMORY: keytab support])

SQUID_CHECK_WORKING_GSSAPI
SQUID_DEFINE_BOOL(HAVE_GSSAPI,$squid_cv_working_gssapi,[GSSAPI support])

SQUID_CHECK_SPNEGO_SUPPORT
SQUID_DEFINE_BOOL(HAVE_SPNEGO,$squid_cv_have_spnego,[SPNEGO support])
```

```
    SQUID_CHECK_WORKING_KRB5
    SQUID_DEFINE_BOOL(HAVE_KRB5,$squid_cv_working_krb5,[KRB5 support])
])


dnl checks whether dbopen needs -ldb to be added to libs
dnl sets ac_cv_dbopen_libdb to either "yes" or "no"

AC_DEFUN([SQUID_CHECK_DBOPEN_NEEDS_LIBDB],[
    AC_CACHE_CHECK(if dbopen needs -ldb,ac_cv_dbopen_libdb, [
        SQUID_STATE_SAVE(dbopen_libdb)
        LIBS="$LIBS -ldb"
        AC_LINK_IFELSE([AC_LANG_PROGRAM([[
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_LIMITS_H
#include <limits.h>
#endif
#if HAVE_DB_185_H
#include <db_185.h>
#elif HAVE_DB_H
#include <db.h>
#endif]],
[[dbopen("", 0, 0, DB_HASH, (void *)0L)]])],
        [ac_cv_dbopen_libdb="yes"],
        [ac_cv_dbopen_libdb="no"])
        SQUID_STATE_ROLLBACK(dbopen_libdb)
    ])
])


dnl check whether regex works by actually compiling one
dnl sets squid_cv_regex_works to either yes or no

AC_DEFUN([SQUID_CHECK_REGEX_WORKS],[
    AC_CACHE_CHECK([if the system-supplied regex lib actually works],squid_cv_regex_works,[
        AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_REGEX_H
#include <regex.h>
```

```
#endif
]], [[
regex_t t; regcomp(&t,"",0);]])],
       [ squid_cv_regex_works=yes ],
       [ squid_cv_regex_works=no ])
   ])
])


AC_DEFUN([SQUID_CHECK_LIBIPHLPAPI],[
   AC_CACHE_CHECK([for libIpHlpApi],squid_cv_have_libiphlpapi,[
      SQUID_STATE_SAVE(iphlpapi)
      LIBS="$LIBS -liphlpapi"
      AC_LINK_IFELSE([AC_LANG_PROGRAM([[
#include <windows.h>
#include <winsock2.h>
#include <iphlpapi.h>
]], [[
   MIB_IPNETTABLE i;
   unsigned long isz=sizeof(i);
   GetIpNetTable(&i,&isz,FALSE);
      ]])],
      [squid_cv_have_libiphlpapi=yes
        SQUID_STATE_COMMIT(iphlpapi)],
      [squid_cv_have_libiphlpapi=no
        SQUID_STATE_ROLLBACK(iphlpapi)])
   ])
   SQUID_STATE_ROLLBACK(iphlpapi)
])

dnl Checks whether the OpenSSL SSL_get_certificate crashes squid and if a
dnl workaround can be used instead of using the SSL_get_certificate
AC_DEFUN([SQUID_CHECK_OPENSSL_GETCERTIFICATE_WORKS],[
   AH_TEMPLATE(SQUID_SSLGETCERTIFICATE_BUGGY, "Define to 1 if the SSL_get_certificate crashes squid")
   AH_TEMPLATE(SQUID_USE_SSLGETCERTIFICATE_HACK, "Define to 1 to use squid workaround for SSL_get_certificate")
   SQUID_STATE_SAVE(check_SSL_get_certificate)
   LIBS="$SSLLIB $LIBS"
   if test "x$SSLLIBDIR" != "x"; then
       LIBS="$LIBS -Wl,-rpath -Wl,$SSLLIBDIR"
   fi

   AC_MSG_CHECKING(whether the SSL_get_certificate is buggy)
```

```
  AC_RUN_IFELSE([
  AC_LANG_PROGRAM(
     [
      #include <openssl/ssl.h>
      #include <openssl/err.h>
     ],
     [
     SSLeay_add_ssl_algorithms();
#if (OPENSSL_VERSION_NUMBER >= 0x10100000L)
     SSL_CTX *sslContext = SSL_CTX_new(TLS_method());
#else
     SSL_CTX *sslContext = SSL_CTX_new(SSLv23_method());
#endif
     SSL *ssl = SSL_new(sslContext);
     X509* cert = SSL_get_certificate(ssl);
     return 0;
     ])
  ],
  [
   AC_MSG_RESULT([no])
  ],
  [
   AC_DEFINE(SQUID_SSLGETCERTIFICATE_BUGGY, 1)
   AC_MSG_RESULT([yes])
  ],
  [
   AC_DEFINE(SQUID_SSLGETCERTIFICATE_BUGGY, 0)
   AC_MSG_RESULT([cross-compile, assuming no])
  ])

  AC_MSG_CHECKING(whether the workaround for SSL_get_certificate works)
  AC_RUN_IFELSE([
  AC_LANG_PROGRAM(
     [
      #include <openssl/ssl.h>
      #include <openssl/err.h>
     ],
     [
     SSLeay_add_ssl_algorithms();
#if (OPENSSL_VERSION_NUMBER >= 0x10100000L)
     SSL_CTX *sslContext = SSL_CTX_new(TLS_method());
#else
     SSL_CTX *sslContext = SSL_CTX_new(SSLv23_method());
#endif
```

```
    X509 ***pCert = (X509 ***)sslContext->cert;
    X509 *sslCtxCert = pCert && *pCert ? **pCert : (X509 *)0x1;
    if (sslCtxCert != NULL)
        return 1;
    return 0;
    ])
  ],
  [
   AC_MSG_RESULT([yes])
   AC_DEFINE(SQUID_USE_SSLGETCERTIFICATE_HACK, 1)
  ],
  [
   AC_MSG_RESULT([no])
  ],
  [
   AC_DEFINE(SQUID_USE_SSLGETCERTIFICATE_HACK, 0)
   AC_MSG_RESULT([cross-compile, assuming no])
  ])

SQUID_STATE_ROLLBACK(check_SSL_get_certificate)
])

dnl Checks whether the    SSL_CTX_new and similar functions require
dnl a const 'SSL_METHOD *' argument
AC_DEFUN([SQUID_CHECK_OPENSSL_CONST_SSL_METHOD],[
  AH_TEMPLATE(SQUID_USE_CONST_SSL_METHOD, "Define to 1 if the SSL_CTX_new and similar
openSSL API functions require 'const SSL_METHOD *'")
  SQUID_STATE_SAVE(check_const_SSL_METHOD)
  AC_MSG_CHECKING(whether SSL_CTX_new and similar openSSL API functions require 'const
SSL_METHOD *'")

  AC_COMPILE_IFELSE([
  AC_LANG_PROGRAM(
    [
     #include <openssl/ssl.h>
     #include <openssl/err.h>
    ],
    [
        const SSL_METHOD *method = NULL;
        SSL_CTX *sslContext = SSL_CTX_new(method);
        return (sslContext != NULL);
    ])
  ],
  [
```

```
    AC_DEFINE(SQUID_USE_CONST_SSL_METHOD, 1)
    AC_MSG_RESULT([yes])
  ],
  [
    AC_MSG_RESULT([no])
  ],
  [])

SQUID_STATE_ROLLBACK(check_const_SSL_METHOD)
]
)

dnl Try to handle TXT_DB related    problems:
dnl 1) The type of TXT_DB::data member changed in openSSL-1.0.1 version
dnl 2) The IMPLEMENT_LHASH_* openSSL macros in openSSL-1.0.1 and later releases is not
dnl       implemented correctly and causes type conversion errors while compiling squid

AC_DEFUN([SQUID_CHECK_OPENSSL_TXTDB],[
  AH_TEMPLATE(SQUID_SSLTXTDB_PSTRINGDATA, "Define to 1 if the TXT_DB uses OPENSSL_PSTRING data member")
  AH_TEMPLATE(SQUID_STACKOF_PSTRINGDATA_HACK, "Define to 1 to use squid workaround for buggy versions of sk_OPENSSL_PSTRING_value")
  AH_TEMPLATE(SQUID_USE_SSLLHASH_HACK, "Define to 1 to use squid workaround for openssl IMPLEMENT_LHASH_* type conversion errors")

  SQUID_STATE_SAVE(check_TXTDB)

  LIBS="$LIBS $SSLLIB"
  squid_cv_check_openssl_pstring="no"
  AC_MSG_CHECKING(whether the TXT_DB use OPENSSL_PSTRING data member)
  AC_COMPILE_IFELSE([
  AC_LANG_PROGRAM(
    [
     #include <openssl/txt_db.h>
    ],
    [
    TXT_DB *db = NULL;
    int i = sk_OPENSSL_PSTRING_num(db->data);
    return 0;
    ])
  ],
  [
   AC_DEFINE(SQUID_SSLTXTDB_PSTRINGDATA, 1)
   AC_MSG_RESULT([yes])
```

```
    squid_cv_check_openssl_pstring="yes"
  ],
  [
    AC_MSG_RESULT([no])
  ],
  [])

  if test x"$squid_cv_check_openssl_pstring" = "xyes"; then
      AC_MSG_CHECKING(whether the squid workaround for buggy versions of
sk_OPENSSL_PSTRING_value should used)
      AC_COMPILE_IFELSE([
      AC_LANG_PROGRAM(
        [
          #include <openssl/txt_db.h>
        ],
        [
        TXT_DB *db = NULL;
        const char ** current_row = ((const char **)sk_OPENSSL_PSTRING_value(db->data, 0));
        return (current_row != NULL);
        ])
      ],
      [
        AC_MSG_RESULT([no])
      ],
      [
        AC_DEFINE(SQUID_STACKOF_PSTRINGDATA_HACK, 1)
        AC_MSG_RESULT([yes])
      ],
      [])
  fi

  AC_MSG_CHECKING(whether the workaround for OpenSSL IMPLEMENT_LHASH_  macros should
used)
  AC_COMPILE_IFELSE([
  AC_LANG_PROGRAM(
    [
      #include <openssl/txt_db.h>

      static unsigned long index_serial_hash(const char **a){}
      static int index_serial_cmp(const char **a, const char **b){}
      static IMPLEMENT_LHASH_HASH_FN(index_serial_hash,const char **)
      static IMPLEMENT_LHASH_COMP_FN(index_serial_cmp,const char **)
    ],
    [
```

```
      TXT_DB *db = NULL;
      TXT_DB_create_index(db, 1, NULL, LHASH_HASH_FN(index_serial_hash),
LHASH_COMP_FN(index_serial_cmp));
      ])
  ],
  [
    AC_MSG_RESULT([no])
  ],
  [
    AC_MSG_RESULT([yes])
    AC_DEFINE(SQUID_USE_SSLLHASH_HACK, 1)
  ],
[])

SQUID_STATE_ROLLBACK(check_TXTDB)
])

dnl Check if we can rewrite the hello message stored in an SSL object.
dnl The tests are very basic, just check if the required members exist in
dnl SSL structure.
AC_DEFUN([SQUID_CHECK_OPENSSL_HELLO_OVERWRITE_HACK],[
  AH_TEMPLATE(SQUID_USE_OPENSSL_HELLO_OVERWRITE_HACK, "Define to 1 if hello message
can be overwritten in SSL struct")
  SQUID_STATE_SAVE(check_openSSL_overwrite_hack)
  AC_MSG_CHECKING(whether hello message can be overwritten in SSL struct)

  AC_COMPILE_IFELSE([
  AC_LANG_PROGRAM(
    [
      #include <openssl/ssl.h>
      #include <openssl/err.h>
      #include <assert.h>
    ],
    [
    SSL *ssl;
    char *random, *msg;
    memcpy(ssl->s3->client_random, random, SSL3_RANDOM_SIZE);
    SSL3_BUFFER *wb=&(ssl->s3->wbuf);
    assert(wb->len == 0);
    memcpy(wb->buf, msg, 0);
    assert(wb->left == 0);
    memcpy(ssl->init_buf->data, msg, 0);
    ssl->init_num = 0;
    ssl->s3->wpend_ret = 0;
```

```
      ssl->s3->wpend_tot = 0;
      ])
   ],
   [
    AC_DEFINE(SQUID_USE_OPENSSL_HELLO_OVERWRITE_HACK, 1)
    AC_MSG_RESULT([yes])
   ],
   [
    AC_MSG_RESULT([no])
   ],
   [])

SQUID_STATE_ROLLBACK(check_openSSL_overwrite_hack)
]
)

dnl check that strnstr() works fine. On Macos X it can cause a buffer overrun
dnl sets squid_cv_func_strnstr to "yes" or "no", and defines HAVE_STRNSTR
AC_DEFUN([SQUID_CHECK_FUNC_STRNSTR],[

# Yay!    This one is    a MacOSX brokenness.    Its not good enough
# to know that strnstr() exists, because MacOSX 10.4 have a bad
# copy that crashes with a buffer over-run!
AH_TEMPLATE(HAVE_STRNSTR,[MacOS brokenness: strnstr() can overrun on that system])
AC_CACHE_CHECK([if strnstr is well implemented], squid_cv_func_strnstr,
   AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
      // we expect this to succeed, or crash on over-run.
      // if it passes otherwise we may need a better check.
int main(int argc, char **argv)
{
      int size = 20;
      char *str = malloc(size);
      memset(str, 'x', size);
      strnstr(str, "fubar", size);
      return 0;
}
   ]])],[squid_cv_func_strnstr="yes"],[squid_cv_func_strnstr="no"],[:])
)
if test "$squid_cv_func_strnstr" = "yes" ; then
   AC_DEFINE(HAVE_STRNSTR,1)
fi
```

```
]) dnl SQUID_CHECK_FUNC_STRNSTR

dnl check that va_copy is implemented and works
dnl sets squid_cv_func_va_copy and defines HAVE_VA_COPY
AC_DEFUN([SQUID_CHECK_FUNC_VACOPY],[

# check that the system provides a functional va_copy call

AH_TEMPLATE(HAVE_VA_COPY, [The system implements a functional va_copy() ])
AC_CACHE_CHECK(if va_copy is implemented, squid_cv_func_va_copy,
   AC_RUN_IFELSE([AC_LANG_SOURCE([[
        #include <stdarg.h>
        #include <stdlib.h>
        int f (int i, ...) {
            va_list args1, args2;
            va_start (args1, i);
            va_copy (args2, args1);
            if (va_arg (args2, int) != 42 || va_arg (args1, int) != 42)
                return 1;
            va_end (args1); va_end (args2);
            return 0;
        }
        int main(int argc, char **argv) { return f (0, 42); }
        ]])],[squid_cv_func_va_copy="yes"],[squid_cv_func_va_copy="no"],[:])
)
if test "$squid_cv_func_va_copy" = "yes" ; then
   AC_DEFINE(HAVE_VA_COPY, 1)
fi

]) dnl SQUID_CHECK_FUNC_VACOPY

dnl same sa SQUID_CHECK_FUNC_VACOPY, but checks __va_copy
dnl sets squid_cv_func___va_copy, and defines HAVE___VA_COPY
AC_DEFUN([SQUID_CHECK_FUNC___VACOPY],[

AH_TEMPLATE(HAVE___VA_COPY,[Some systems have __va_copy instead of va_copy])
AC_CACHE_CHECK(if __va_copy is implemented, squid_cv_func___va_copy,
   AC_RUN_IFELSE([AC_LANG_SOURCE([[
        #include <stdarg.h>
        #include <stdlib.h>
        int f (int i, ...) {
            va_list args1, args2;
            va_start (args1, i);
```

```
        __va_copy (args2, args1);
        if (va_arg (args2, int) != 42 || va_arg (args1, int) != 42)
          return 1;
        va_end (args1); va_end (args2);
        return 0;
      }
      int main(int argc, char **argv) { return f (0, 42); }
      ]])],[squid_cv_func___va_copy="yes"],[squid_cv_func___va_copy="no"],[:])
)
if test "$squid_cv_func___va_copy" = "yes" ; then
  AC_DEFINE(HAVE___VA_COPY, 1)
fi
]) dnl SQUID_CHECK_FUNC___VACOPY


dnl check that epoll actually works
dnl sets squid_cv_epoll_works to "yes" or "no"
AC_DEFUN([SQUID_CHECK_EPOLL],[

    AC_CACHE_CHECK(if epoll works, squid_cv_epoll_works,
      AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <sys/epoll.h>
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char **argv)
{
    int fd = epoll_create(256);
    if (fd < 0) {
    perror("epoll_create:");
    return 1;
    }
    return 0;
}
      ]])],[squid_cv_epoll_works=yes],[squid_cv_epoll_works=no],[:]))

]) dnl SQUID_CHECK_EPOLL

dnl check that /dev/poll actually works
dnl sets squid_cv_devpoll_works to "yes" or "no"
AC_DEFUN([SQUID_CHECK_DEVPOLL],[

    AC_CACHE_CHECK(if /dev/poll works, squid_cv_devpoll_works,
      AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <sys/devpoll.h>
```

```
#include <fcntl.h>
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char **argv)
{
       int fd = open("/dev/poll", O_RDWR);
       if (fd < 0) {
            perror("devpoll_create:");
            return 1;
       }
       return 0;
}
          ])],[squid_cv_devpoll_works=yes],[squid_cv_devpoll_works=no],[:]))

]) dnl SQUID_CHECK_DEVPOLL


dnl check that we have functional libcap2 headers
dnl sets squid_cv_sys_capability_works to "yes" or "no"

AC_DEFUN([SQUID_CHECK_FUNCTIONAL_LIBCAP2],[
   AC_CACHE_CHECK([for operational libcap2 headers],
                          squid_cv_sys_capability_works,
      AC_LINK_IFELSE([AC_LANG_PROGRAM([[
#include <stdlib.h>
#include <stddef.h>
#include <sys/capability.h>
]], [[
       capget(NULL, NULL);
       capset(NULL, NULL);
]])],
     [squid_cv_sys_capability_works=yes],
     [squid_cv_sys_capability_works=no])
   )
])


dnl From Samba. Thanks!
dnl check that we have Unix sockets. Sets squid_cv_unixsocket to either yes or no depending on the
check

AC_DEFUN([SQUID_CHECK_UNIX_SOCKET],[
   AC_CACHE_CHECK([for unix domain sockets],squid_cv_unixsocket, [
         AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
```

```
#include <sys/types.h>
#include <stdlib.h>
#include <stddef.h>
#include <sys/socket.h>
#include <sys/un.h>]], [[
   struct sockaddr_un sunaddr;
   sunaddr.sun_family = AF_UNIX;
   ]])],[squid_cv_unixsocket=yes],[squid_cv_unixsocket=no])])
])


dnl check the default FD_SETSIZE size.
dnl not cached, people are likely to tune this
dnl defines DEFAULT_FD_SETSIZE

AC_DEFUN([SQUID_CHECK_DEFAULT_FD_SETSIZE],[
AC_MSG_CHECKING(Default FD_SETSIZE value)
AC_RUN_IFELSE([AC_LANG_SOURCE([[
#if HAVE_STDIO_H
#include <stdio.h>
#endif
#if HAVE_UNISTD_H
#include <unistd.h>
#endif
#if HAVE_STDLIB_H
#include <stdlib.h>
#endif
#if HAVE_SYS_TIME_H
#include <sys/time.h>
#endif
#if HAVE_SYS_SELECT_H
#include <sys/select.h>
#endif
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
#endif
int main(int argc, char **argv) {
      FILE *fp = fopen("conftestval", "w");
      fprintf (fp, "%d\n", FD_SETSIZE);
```

```
        return 0;
}
]])],[DEFAULT_FD_SETSIZE=`cat
conftestval`],[DEFAULT_FD_SETSIZE=256],[DEFAULT_FD_SETSIZE=256])
AC_MSG_RESULT($DEFAULT_FD_SETSIZE)
AC_DEFINE_UNQUOTED(DEFAULT_FD_SETSIZE, $DEFAULT_FD_SETSIZE, [Default FD_SETSIZE
value])
])


dnl checks the maximum number of filedescriptor we can open
dnl sets shell var squid_filedescriptors_num

AC_DEFUN([SQUID_CHECK_MAXFD],[
AC_CHECK_FUNCS(setrlimit)
AC_MSG_CHECKING(Maximum number of filedescriptors we can open)
dnl damn! FreeBSD pthreads break dup2().
SQUID_STATE_SAVE(maxfd)
   case $host in
   i386-unknown-freebsd*)
        if echo "$LDFLAGS" | grep -q pthread; then
      LDFLAGS=`echo $LDFLAGS | sed -e "s/-pthread//"`
        fi
   esac
   AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/time.h> /* needed on FreeBSD */
#include <sys/param.h>
#include <sys/resource.h>
int main(int argc, char **argv) {
        FILE *fp;
        int i,j;
#if defined(__CYGWIN32__) || defined (__CYGWIN__)
        /* getrlimit and sysconf returns bogous values on cygwin32.
         * Number of fds is virtually unlimited in cygwin (sys/param.h)
         * __CYGWIN32__ is deprecated.
         */
        i = NOFILE;
#else
#if HAVE_SETRLIMIT
        struct rlimit rl;
#if defined(RLIMIT_NOFILE)
```

```
    if (getrlimit(RLIMIT_NOFILE, &rl) < 0) {
        perror("getrlimit: RLIMIT_NOFILE");
    } else {
#if defined(__APPLE__)
        /* asking for more than OPEN_MAX fails on Leopard */
        rl.rlim_cur = (OPEN_MAX < rl.rlim_max ? OPEN_MAX : rl.rlim_max);
#else
        rl.rlim_cur = rl.rlim_max;          /* set it to the max */
#endif
        if (setrlimit(RLIMIT_NOFILE, &rl) < 0) {
            perror("setrlimit: RLIMIT_NOFILE");
        }
    }
#elif defined(RLIMIT_OFILE)
    if (getrlimit(RLIMIT_OFILE, &rl) < 0) {
        perror("getrlimit: RLIMIT_OFILE");
    } else {
        rl.rlim_cur = rl.rlim_max;          /* set it to the max */
        if (setrlimit(RLIMIT_OFILE, &rl) < 0) {
            perror("setrlimit: RLIMIT_OFILE");
        }
    }
#endif /* RLIMIT_NOFILE */
#endif /* HAVE_SETRLIMIT */
    /* by starting at 2^14, we will never get higher
    than 2^15 for squid_filedescriptors_num */
        i = j = 1<<14;
        while (j) {
                j >>= 1;
                if (dup2(0, i) < 0) {
                        i -= j;
                } else {
                        close(i);
                        i += j;
                }
        }
        i++;
#endif /* IF !DEF CYGWIN */
    fp = fopen("conftestval", "w");
    fprintf (fp, "%d\n", i & ~0x3F);
    return 0;
}
   ]])],[squid_filedescriptors_num=`cat
conftestval`],[squid_filedescriptors_num=256],[squid_filedescriptors_num=256])
```

```
  dnl Microsoft MSVCRT.DLL supports 2048 maximum FDs
  case "$host_os" in
  mingw|mingw32)
    squid_filedescriptors_num="2048"
    ;;
  esac
  AC_MSG_RESULT($squid_filedescriptors_num)
SQUID_STATE_ROLLBACK(maxfd)

if test `expr $squid_filedescriptors_num % 64` != 0; then
    AC_MSG_WARN([$squid_filedescriptors_num is not an multiple of 64. This may cause issues on certain platforms.])
fi
])


dnl Check whether this OS defines sin6_len as a member of sockaddr_in6 as a backup to ss_len
dnl defines HAVE_SIN6_LEN_IN_SAI
dnl TODO: move to AC_CHECK_MEMBER?

AC_DEFUN([SQUID_CHECK_SIN6_LEN_IN_SAI],[
AC_CACHE_CHECK([for sin6_len field in struct sockaddr_in6],
                ac_cv_have_sin6_len_in_struct_sai, [
    AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
                ]], [[ struct sockaddr_in6 s; s.sin6_len = 1; ]])],[ ac_cv_have_sin6_len_in_struct_sai="yes" ],[ ac_cv_have_sin6_len_in_struct_sai="no"
    ])
])
SQUID_DEFINE_BOOL(HAVE_SIN6_LEN_IN_SAI,$ac_cv_have_sin6_len_in_struct_sai,
    [Defined if struct sockaddr_in6 has sin6_len])
])


dnl Check whether this OS defines ss_len as a member of sockaddr_storage
dnl defines HAVE_SS_LEN_IN_SS
dnl TODO: move to AC_CHECK_MEMBER?

AC_DEFUN([SQUID_CHECK_SS_LEN_IN_SOCKADDR_STORAGE],[
AC_CACHE_CHECK([for ss_len field in struct sockaddr_storage],
        ac_cv_have_ss_len_in_struct_ss, [
  AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
```

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
        ]], [[ struct sockaddr_storage s; s.ss_len =
1; ]])],[ ac_cv_have_ss_len_in_struct_ss="yes" ],[ ac_cv_have_ss_len_in_struct_ss="no"
    ])
])
SQUID_DEFINE_BOOL(HAVE_SS_LEN_IN_SS,$ac_cv_have_ss_len_in_struct_ss,
  [Define if sockaddr_storage has field ss_len])
])


dnl Check whether this OS defines sin_len as a member of sockaddr_in as a backup to ss_len
dnl defines HAVE_SIN_LEN_IN_SAI
dnl TODO: move to AC_CHECK_MEMBER?

AC_DEFUN([SQUID_CHECK_SIN_LEN_IN_SOCKADDR_IN],[
AC_CACHE_CHECK([for sin_len field in struct sockaddr_in],
                ac_cv_have_sin_len_in_struct_sai, [
        AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
                ]], [[ struct sockaddr_in s; s.sin_len =
1; ]])],[ ac_cv_have_sin_len_in_struct_sai="yes" ],[ ac_cv_have_sin_len_in_struct_sai="no"
        ])
])
SQUID_DEFINE_BOOL(HAVE_SIN_LEN_IN_SAI,$ac_cv_have_sin_len_in_struct_sai,[Define if
sockaddr_in has field sin_len])
])


dnl detects default UDP buffer size
dnl not cached since people are likely to tune this
dnl defines SQUID_DETECT_UDP_SO_SNDBUF

AC_DEFUN([SQUID_DETECT_UDP_SND_BUFSIZE],[
AC_MSG_CHECKING(Default UDP send buffer size)
AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#if HAVE_SYS_SOCKET_H
#include <sys/socket.h>
```

```
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
#endif
int main(int argc, char **argv)
{
      FILE *fp;
            int fd,val=0;
#if  (defined(WIN32)  ||  defined(__WIN32__)  ||  defined(__WIN32))  &&  !(defined(__CYGWIN32__)  ||
defined(__CYGWIN__))
            int len=sizeof(int);
      WSADATA wsaData;
      WSAStartup(2, &wsaData);
#else
            socklen_t len=sizeof(socklen_t);
#endif
      if ((fd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) return 1;
#if  (defined(WIN32)  ||  defined(__WIN32__)  ||  defined(__WIN32))  &&  !(defined(__CYGWIN32__)  ||
defined(__CYGWIN__))
            if (getsockopt(fd, SOL_SOCKET, SO_SNDBUF, (char *)&val, &len) < 0) return 1;
      WSACleanup();
#else
            if (getsockopt(fd, SOL_SOCKET, SO_SNDBUF, &val, &len) < 0) return 1;
#endif
      if (val<=0) return 1;
            fp = fopen("conftestval", "w");
            fprintf (fp, "%d\n", val);
      return 0;
}
]])],[SQUID_DETECT_UDP_SO_SNDBUF=`cat
conftestval`],[SQUID_DETECT_UDP_SO_SNDBUF=16384],[SQUID_DETECT_UDP_SO_SNDBUF=163
84])
AC_MSG_RESULT($SQUID_DETECT_UDP_SO_SNDBUF)
AC_DEFINE_UNQUOTED(SQUID_DETECT_UDP_SO_SNDBUF,
$SQUID_DETECT_UDP_SO_SNDBUF,[UDP send buffer size])
])


dnl detects default UDP buffer size
```

```
dnl not cached since people are likely to tune this
dnl defines SQUID_DETECT_UDP_SO_RCVBUF

AC_DEFUN([SQUID_DETECT_UDP_RECV_BUFSIZE],[
AC_MSG_CHECKING(Default UDP receive buffer size)
AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#if HAVE_SYS_SOCKET_H
#include <sys/socket.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
#endif
int main(int argc, char **argv)
{
      FILE *fp;
            int fd,val=0;
#if  (defined(WIN32)  ||  defined(__WIN32__)  ||  defined(__WIN32))  &&  !(defined(__CYGWIN32__)  ||
defined(__CYGWIN__))
            int len=sizeof(int);
      WSADATA wsaData;
      WSAStartup(2, &wsaData);
#else
            socklen_t len=sizeof(socklen_t);
#endif
      if ((fd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) return 1;
#if  (defined(WIN32)  ||  defined(__WIN32__)  ||  defined(__WIN32))  &&  !(defined(__CYGWIN32__)  ||
defined(__CYGWIN__))
            if (getsockopt(fd, SOL_SOCKET, SO_RCVBUF, (char *)&val, &len) < 0) return 1;
      WSACleanup();
#else
            if (getsockopt(fd, SOL_SOCKET, SO_RCVBUF, &val, &len) < 0) return 1;
#endif
      if (val <= 0) return 1;
      fp = fopen("conftestval", "w");
      fprintf (fp, "%d\n", val);
      return 0;
```

```
}
]])],[SQUID_DETECT_UDP_SO_RCVBUF=`cat
conftestval`],[SQUID_DETECT_UDP_SO_RCVBUF=16384],[SQUID_DETECT_UDP_SO_RCVBUF=163
84])
AC_MSG_RESULT($SQUID_DETECT_UDP_SO_RCVBUF)
AC_DEFINE_UNQUOTED(SQUID_DETECT_UDP_SO_RCVBUF,
$SQUID_DETECT_UDP_SO_RCVBUF,[UDP receive buffer size])
])


dnl detects default TCP buffer size
dnl not cached since people are likely to tune this
dnl defines SQUID_TCP_SO_SNDBUF

AC_DEFUN([SQUID_DETECT_TCP_SND_BUFSIZE],[
AC_MSG_CHECKING(Default TCP send buffer size)
AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#if HAVE_SYS_SOCKET_H
#include <sys/socket.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
#endif
int main(int argc, char **argv)
{
      FILE *fp;
            int fd,val=0;
#if  (defined(WIN32)  ||  defined(__WIN32__)  ||  defined(__WIN32))  &&  !(defined(__CYGWIN32__)  ||
defined(__CYGWIN__))
            int len=sizeof(int);
      WSADATA wsaData;
      WSAStartup(2, &wsaData);
#else
            socklen_t len=sizeof(socklen_t);
#endif
      if ((fd = socket(AF_INET, SOCK_STREAM, 0)) < 0) return 1;
```

```
#if (defined(WIN32) || defined(__WIN32__) || defined(__WIN32)) && !(defined(__CYGWIN32__) || defined(__CYGWIN__))
        if (getsockopt(fd, SOL_SOCKET, SO_SNDBUF, (char *)&val, &len) < 0) return 1;
    WSACleanup();
#else
        if (getsockopt(fd, SOL_SOCKET, SO_SNDBUF, &val, &len) < 0) return 1;
#endif
    if (val <= 0) return 1;
    fp = fopen("conftestval", "w");
    fprintf (fp, "%d\n", val);
    return 0;
}
])],[SQUID_TCP_SO_SNDBUF=`cat conftestval`],[SQUID_TCP_SO_SNDBUF=16384],[SQUID_TCP_SO_SNDBUF=16384])
AC_MSG_RESULT($SQUID_TCP_SO_SNDBUF)
if test $SQUID_TCP_SO_SNDBUF -gt 32768; then
    AC_MSG_NOTICE([Limiting send buffer size to 32K])
    SQUID_TCP_SO_SNDBUF=32768
fi
AC_DEFINE_UNQUOTED(SQUID_TCP_SO_SNDBUF, $SQUID_TCP_SO_SNDBUF,[TCP send buffer size])
])


dnl detects default TCP buffer size
dnl not cached since people are likely to tune this
dnl defines SQUID_TCP_SO_RECVBUF

AC_DEFUN([SQUID_DETECT_TCP_RECV_BUFSIZE],[
AC_MSG_CHECKING(Default TCP receive buffer size)
AC_RUN_IFELSE([AC_LANG_SOURCE([[
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#if HAVE_SYS_SOCKET_H
#include <sys/socket.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
```

```
#endif
int main(int argc, char **argv)
{
	FILE *fp;
		int fd,val=0;
#if (defined(WIN32) || defined(__WIN32__) || defined(__WIN32)) && !(defined(__CYGWIN32__) || defined(__CYGWIN__))
		int len=sizeof(int);
	WSADATA wsaData;
	WSAStartup(2, &wsaData);
#else
		socklen_t len=sizeof(socklen_t);
#endif
	if ((fd = socket(AF_INET, SOCK_STREAM, 0)) < 0) return 1;
#if (defined(WIN32) || defined(__WIN32__) || defined(__WIN32)) && !(defined(__CYGWIN32__) || defined(__CYGWIN__))
		if (getsockopt(fd, SOL_SOCKET, SO_RCVBUF, (char *)&val, &len) < 0) return 1;
	WSACleanup();
#else
		if (getsockopt(fd, SOL_SOCKET, SO_RCVBUF, &val, &len) < 0) return 1;
#endif
	if (val <= 0) return 1;
	fp = fopen("conftestval", "w");
	fprintf (fp, "%d\n", val);
	return 0;
}
]])],[SQUID_TCP_SO_RCVBUF=`cat
conftestval`],[SQUID_TCP_SO_RCVBUF=16384],[SQUID_TCP_SO_RCVBUF=16384])
AC_MSG_RESULT($SQUID_TCP_SO_RCVBUF)
if test $SQUID_TCP_SO_RCVBUF -gt 65535; then
	AC_MSG_NOTICE([Limiting receive buffer size to 64K])
	SQUID_TCP_SO_RCVBUF=65535
fi
AC_DEFINE_UNQUOTED(SQUID_TCP_SO_RCVBUF, $SQUID_TCP_SO_RCVBUF,[TCP receive
buffer size])
])


dnl check if we need to define sys_errlist as external
dnl defines NEED_SYS_ERRLIST

AC_DEFUN([SQUID_CHECK_NEED_SYS_ERRLIST],[
AC_CACHE_CHECK(if sys_errlist is already defined, ac_cv_needs_sys_errlist,
  AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[#include <stdio.h>]], [[char *s =
```

```
sys_errlist;]])],[ac_cv_needs_sys_errlist="no"],[ac_cv_needs_sys_errlist="yes"])
)
SQUID_DEFINE_BOOL(NEED_SYS_ERRLIST,$ac_cv_needs_sys_errlist,[If we need to declare sys_errlist as extern])
])


dnl check if MAXPATHLEN is defined in the system headers
dnl or define it ourselves

AC_DEFUN([SQUID_CHECK_MAXPATHLEN],[
AC_MSG_CHECKING(for system-provided MAXPATHLEN)
AC_LINK_IFELSE([
   AC_LANG_PROGRAM([[
#include <sys/param.h>]], [[
int i = MAXPATHLEN;]])], [
   AC_MSG_RESULT(yes)], [
   AC_MSG_RESULT(no)
   AC_DEFINE(MAXPATHLEN,256,[If MAXPATHLEN has not been defined])])
])


dnl check that we have a working statvfs
dnl sets the ac_cv_func_statvfs shell variable and defines HAVE_STATVFS

AC_DEFUN([SQUID_CHECK_WORKING_STATVFS],[
AC_CACHE_CHECK(for working statvfs() interface,ac_cv_func_statvfs,[
   AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/statvfs.h>
]], [[
struct statvfs sfs;
sfs.f_blocks = sfs.f_bfree = sfs.f_frsize =
sfs.f_files = sfs.f_ffree = 0;
statvfs("/tmp", &sfs);
]])],[ac_cv_func_statvfs=yes],[ac_cv_func_statvfs=no])
])
SQUID_DEFINE_BOOL(HAVE_STATVFS,$ac_cv_func_statvfs,[set to 1 if our system has statvfs(), and if it actually works])
])
```

```
dnl Check whether this OS defines f_frsize as a member of struct statfs
AC_DEFUN([SQUID_CHECK_F_FRSIZE_IN_STATFS],[
AC_CACHE_CHECK([for f_frsize field in struct statfs],
                        ac_cv_have_f_frsize_in_struct_statfs, [
            AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#if HAVE_SYS_STATFS_H
#include <sts/statfs.h>
#endif
#if HAVE_SYS_STATVFS_H
#include <sts/statvfs.h>
#endif
#if HAVE_SYS_VFS_H
#include <sts/vfs.h>
#endif
                        ]], [[ struct statfs s; s.f_frsize =
0; ]])],[ ac_cv_have_f_frsize_in_struct_statfs="yes" ],[ ac_cv_have_f_frsize_in_struct_statfs="no"
            ])
])
SQUID_DEFINE_BOOL(HAVE_F_FRSIZE_IN_STATFS,$ac_cv_have_f_frsize_in_struct_statfs,[Define  if
struct statfs has field f_frsize (Linux 2.6 or later)])
])


dnl check that we can use the libresolv _dns_ttl_ hack
dnl sets the ac_cv_libresolv_dns_ttl_hack shell variable and defines LIBRESOLV_DNS_TTL_HACK

AC_DEFUN([SQUID_CHECK_LIBRESOLV_DNS_TTL_HACK],[
   AC_CACHE_CHECK(for libresolv _dns_ttl_ hack, ac_cv_libresolv_dns_ttl_hack, [
     AC_LINK_IFELSE([AC_LANG_PROGRAM([[extern int _dns_ttl_;]], [[return _dns_ttl_;]])],
        [ac_cv_libresolv_dns_ttl_hack=yes],[ac_cv_libresolv_dns_ttl_hack=no]) ])
   SQUID_DEFINE_BOOL(LIBRESOLV_DNS_TTL_HACK,$ac_cv_libresolv_dns_ttl_hack,
        [libresolv.a has been hacked to export _dns_ttl_])
])


dnl checks for availability of some resolver fields
dnl sets ac_cv_have_res_ext_nsaddr_list shell variable
dnl defines _SQUID_RES_NSADDR6_COUNT _SQUID_RES_NSADDR6_LARRAY
dnl defines _SQUID_RES_NSADDR6_LPTR _SQUID_RES_NSADDR6_COUNT
dnl defines _SQUID_RES_NSADDR_LIST _SQUID_RES_NSADDR_COUNT

AC_DEFUN([SQUID_CHECK_RESOLVER_FIELDS],[
   AC_CACHE_CHECK(for _res_ext.nsaddr_list, ac_cv_have_res_ext_nsaddr_list,
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
```

```
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_ARPA_INET_H
#include <arpa/inet.h>
#endif
#if HAVE_ARPA_NAMESER_H
#include <arpa/nameser.h>
#endif
#if HAVE_RESOLV_H
#include <resolv.h>
#endif
      ]],
      [[_res_ext.nsaddr_list[[0]].s_addr;]])],[
         ac_cv_have_res_ext_nsaddr_list="yes" ],[
         ac_cv_have_res_ext_nsaddr_list="no"]))
   if test "$ac_cv_have_res_ext_nsaddr_list" = "yes" ; then
      AC_DEFINE(_SQUID_RES_NSADDR6_LARRAY,_res_ext.nsaddr_list,[If  _res_ext  structure  has
nsaddr_list member])
      AC_DEFINE(_SQUID_RES_NSADDR6_COUNT,ns6count,[Nameserver Counter for IPv6 _res_ext])
   fi

if test "$_SQUID_RES_NSADDR6_LIST" = ""; then
   AC_CACHE_CHECK(for _res._u._ext.nsaddrs, ac_cv_have_res_ext_nsaddrs,
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_ARPA_INET_H
#include <arpa/inet.h>
#endif
#if HAVE_ARPA_NAMESER_H
#include <arpa/nameser.h>
#endif
#if HAVE_RESOLV_H
#include <resolv.h>
#endif
      ]], i
```

```
    [[_res._u._ext.nsaddrs[[0]]->sin6_addr;]])],
    [ac_cv_have_res_ext_nsaddrs="yes"],[ac_cv_have_res_ext_nsaddrs="no"])
  if test "$ac_cv_have_res_ext_nsaddrs" = "yes" ; then
    AC_DEFINE(_SQUID_RES_NSADDR6_LPTR,_res._u._ext.nsaddrs,[If _res structure has _ext.nsaddrs member])
    AC_DEFINE(_SQUID_RES_NSADDR6_COUNT,_res._u._ext.nscount6,[Nameserver Counter for IPv6 _res])
  fi
fi

AC_CACHE_CHECK(for _res.nsaddr_list, ac_cv_have_res_nsaddr_list,
  AC_COMPILE_IFELSE([
    AC_LANG_PROGRAM([[
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_ARPA_INET_H
#include <arpa/inet.h>
#endif
#if HAVE_ARPA_NAMESER_H
#include <arpa/nameser.h>
#endif
#if HAVE_RESOLV_H
#include <resolv.h>
#endif
  ]], [[_res.nsaddr_list[[0]];]])],
  [ac_cv_have_res_nsaddr_list="yes"],[ac_cv_have_res_nsaddr_list="no"])
  if test $ac_cv_have_res_nsaddr_list = "yes" ; then
    AC_DEFINE(_SQUID_RES_NSADDR_LIST,_res.nsaddr_list,[If _res structure has nsaddr_list member])
    AC_DEFINE(_SQUID_RES_NSADDR_COUNT,_res.nscount,[Nameserver counter for IPv4 _res])
  fi

  if test "$_SQUID_RES_NSADDR_LIST" = ""; then
    AC_CACHE_CHECK(for _res.ns_list, ac_cv_have_res_ns_list,
    AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
```

```
#endif
#if HAVE_ARPA_INET_H
#include <arpa/inet.h>
#endif
#if HAVE_ARPA_NAMESER_H
#include <arpa/nameser.h>
#endif
#if HAVE_RESOLV_H
#include <resolv.h>
#endif
   ]],
   [[_res.ns_list[[0]].addr;]])],
   [ac_cv_have_res_ns_list="yes"],[ac_cv_have_res_ns_list="no"]))
   if test $ac_cv_have_res_ns_list = "yes" ; then
      AC_DEFINE(_SQUID_RES_NSADDR_LIST,_res.ns_list,[If _res structure has ns_list member])
      AC_DEFINE(_SQUID_RES_NSADDR_COUNT,_res.nscount,[Nameserver counter for IPv4 _res])
   fi
fi
])


dnl checks the winsock library to use (ws2_32 or wsock32)
dnl may set ac_cv_func_select as a side effect
AC_DEFUN([SQUID_CHECK_WINSOCK_LIB],[
   AC_CHECK_HEADERS(winsock2.h winsock.h)
   SQUID_STATE_SAVE(winsock)
   SQUID_SEARCH_LIBS([squid_getprotobynumber],[ws2_32 wsock32],,,,[
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
#endif
/* ugly hack. */
void squid_getprotobynumber(void) {
      getprotobynumber(1);
}
   ])
   AC_MSG_CHECKING([for winsock library])
   case "$ac_cv_search_squid_getprotobynumber" in
      "no")
         AC_MSG_RESULT([winsock library not found])
         ;;
      "none required")
         AC_MSG_RESULT([winsock library already in LIBS])
```

```
        ;;
      "-lws2_32")
        AC_MSG_RESULT([winsock2])
        XTRA_LIBS="-lws2_32 $XTRA_LIBS"
        ac_cv_func_select='yes'
        ;;
      "-lwsock32")
        AC_MSG_RESULT([winsock])
        XTRA_LIBS="-lwsock32 $XTRA_LIBS"
        ac_cv_func_select='yes'
        ;;
  esac
  SQUID_STATE_ROLLBACK(winsock)
])

dnl check that setresuid is properly implemented.
dnl sets squid_cv_resuid_works to "yes" or "no"
AC_DEFUN([SQUID_CHECK_SETRESUID_WORKS],[
  AC_CACHE_CHECK(if setresuid is actually implemented, squid_cv_resuid_works,
    AC_RUN_IFELSE([
      AC_LANG_SOURCE([[
#if HAVE_STDLIB_H
#include <stdlib.h>
#endif
#if HAVE_STDIO_H
#include <stdio.h>
#endif
#if HAVE_UNISTD_H
#include <unistd.h>
#endif
  int main(int argc, char **argv) {
    if(setresuid(-1,-1,-1)) {
      perror("setresuid:");
      return 1;
    }
    return 0;
  }
  ]])],[
    squid_cv_resuid_works="yes" ],[
    squid_cv_resuid_works="no" ],[:])
  )
])

dnl check that we have functional CPU clock access for the profiler
```

```
dnl sets squid_cv_profiler_works to "yes" or "no"

AC_DEFUN([SQUID_CHECK_FUNCTIONAL_CPU_PROFILER],[
   AC_CACHE_CHECK([for operational CPU clock access],
                        squid_cv_cpu_profiler_works,
      AC_PREPROC_IFELSE([AC_LANG_SOURCE([[
#if defined(__GNUC__) && ( defined(__i386) || defined(__i386__) )
// okay
#elif defined(__GNUC__) && ( defined(__x86_64) || defined(__x86_64__) )
// okay
#elif defined(__GNUC__) && defined(__alpha)
// okay
#elif defined(_M_IX86) && defined(_MSC_VER) /* x86 platform on Microsoft C Compiler ONLY */
// okay
#else
#error This CPU is unsupported. No profiling available here.
#endif
   ]])],[
   squid_cv_cpu_profiler_works=yes],[
   squid_cv_cpu_profiler_works=no])
   )
])

dnl check whether recv takes a char* or void* as a second argument
AC_DEFUN([SQUID_CHECK_RECV_ARG_TYPE],[
   AC_CACHE_CHECK([whether recv takes a pointer to void or char as second argument],
            squid_cv_recv_second_arg_type, [
                        AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <sys/types.h>
#if HAVE_SYS_SOCKET_H
#include <sys/socket.h>
#endif
#if HAVE_WINSOCK2_H
#include <winsock2.h>
#elif HAVE_WINSOCK_H
#include <winsock.h>
#endif
int main (int argc, char ** argv) {
           void *buf;
   recv(0,buf,0,0);
}
]])],[squid_cv_recv_second_arg_type=void],
        [squid_cv_recv_second_arg_type=char])
   AC_MSG_RESULT($squid_cv_recv_second_arg_type*)
```

```
  ])
  AC_DEFINE_UNQUOTED(RECV_ARG_TYPE,$squid_cv_recv_second_arg_type,
    [Base type of the second argument to recv(2)])
])


dnl check whether Solaris has broken IPFilter headers (Solaris 10 at least does)
AC_DEFUN([SQUID_CHECK_BROKEN_SOLARIS_IPFILTER],[
  if test "x$squid_cv_broken_ipfilter_minor_t" = "x"; then
    AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#       include <sys/types.h>
#       include <sys/time.h>
#       include <sys/ioccom.h>
#       include <netinet/in.h>

#       include <netinet/ip_compat.h>
#       include <netinet/ip_fil.h>
#       include <netinet/ip_nat.h>
    ]])],[
      AC_MSG_RESULT(no)
      squid_cv_broken_ipfilter_minor_t=0
    ],[
      ## on fail, test the hack
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#define minor_t fubaar
#           include <sys/types.h>
#           include <sys/time.h>
#           include <sys/ioccom.h>
#           include <netinet/in.h>
#undef minor_t
#           include <netinet/ip_compat.h>
#           include <netinet/ip_fil.h>
#           include <netinet/ip_nat.h>
        ]])],[
          AC_MSG_RESULT(yes)
          squid_cv_broken_ipfilter_minor_t=1
        ],[
          AC_MSG_ERROR(unable to make IPFilter work with netinet/ headers)
        ])
    ])
  fi


AC_DEFINE_UNQUOTED(USE_SOLARIS_IPFILTER_MINOR_T_HACK,$squid_cv_broken_ipfilter_min
```

```
or_t,
      [Workaround IPFilter minor_t breakage])

## check for IPFilter headers that require this hack
## (but first netinet/in.h and sys/ioccom.h which they depend on)
   AC_CHECK_HEADERS( \
      netinet/in.h \
      sys/ioccom.h \
      ip_compat.h \
      ip_fil_compat.h \
      ip_fil.h \
      ip_nat.h \
      netinet/ip_compat.h \
      netinet/ip_fil_compat.h \
      netinet/ip_fil.h \
      netinet/ip_nat.h \
   ,,,[
#if USE_SOLARIS_IPFILTER_MINOR_T_HACK
#define minor_t fubar
#endif
#if HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif
#if HAVE_SYS_TIME_H
#include <sys/time.h>
#endif
#if HAVE_NETINET_IN_H
#include <netinet/in.h>
#endif
#if HAVE_SYS_IOCCOM_H
#include <sys/ioccom.h>
#endif
#if USE_SOLARIS_IPFILTER_MINOR_T_HACK
#undef minor_t
#endif
#if HAVE_IP_COMPAT_H
#include <ip_compat.h>
#elif HAVE_NETINET_IP_COMPAT_H
#include <netinet/ip_compat.h>
#endif
#if HAVE_IP_FIL_H
#include <ip_fil.h>
#elif HAVE_NETINET_IP_FIL_H
#include <netinet/ip_fil.h>
```

```
#endif
#if !defined(IPFILTER_VERSION)
#define IPFILTER_VERSION		5000004
#endif
   ])
])

dnl check whether PAM's struct pam_conv takes a const (linux-style) or
dnl non-const (solaris-style) parametrs to the conv function.
dnl
dnl sets the shell variable squid_cv_pam_conv_signature to either
dnl "linux", "solaris" or "unknown".
dnl defines the C preprocessor macro PAM_CONV_FUNC_CONST_PARM to either
dnl "static" (linux-style) or the empty string (solaris-style or default)

AC_DEFUN([CHECK_STRUCT_PAM_CONV], [
   AH_TEMPLATE([PAM_CONV_FUNC_CONST_PARM],
      [Defined to const or empty depending on the style used by the OS to refer to the PAM message dialog
func])
   AC_CACHE_CHECK([for PAM conversation struct signature type],
                  squid_cv_pam_conv_signature, [
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <security/pam_appl.h>
static int
password_conversation(int num_msg, const struct pam_message **msg, struct pam_response **resp,
void *appdata_ptr) {}
static struct pam_conv conv = { &password_conversation, 0 };
]])], [
     squid_cv_pam_conv_signature=linux
], [
      AC_COMPILE_IFELSE([AC_LANG_PROGRAM([[
#include <security/pam_appl.h>
static int
password_conversation(int num_msg, struct pam_message **msg, struct pam_response **resp, void
*appdata_ptr) {}
static struct pam_conv conv = { &password_conversation, 0 };
]])], [
   squid_cv_pam_conv_signature=solaris
 ], [
   squid_cv_pam_conv_signature=unknown
   ])
      ])
   ])
   case $squid_cv_pam_conv_signature in
```

```
      linux) AC_DEFINE([PAM_CONV_FUNC_CONST_PARM],[const]) ;;
      solaris) AC_DEFINE([PAM_CONV_FUNC_CONST_PARM],[]) ;;
      *) AC_DEFINE([PAM_CONV_FUNC_CONST_PARM],[]) ;;
   esac
]) dnl CHECK_STRUCT_PAM_CONV


dnl save main environment variables to variables to the namespace defined by the
dnl first argument (prefix)
dnl e.g. SQUID_SAVEFLAGS([foo]) will save CFLAGS to foo_CFLAGS etc.
dnl Saved variables are:
dnl CFLAGS, CXXFLAGS, LDFLAGS, LIBS plus any variables specified as
dnl second argument
AC_DEFUN([SQUID_STATE_SAVE],[
# save state, key is $1
$1_CFLAGS="${CFLAGS}"
$1_CXXFLAGS="${CXXFLAGS}"
$1_LDFLAGS="${LDFLAGS}"
$1_LIBS="${LIBS}"
$1_CC="${CC}"
$1_CXX="${CXX}"
$1_squid_saved_vars="$2"
for squid_util_var_tosave in $$1_squid_saved_vars
do
      squid_util_var_tosave2="$1_${squid_util_var_tosave}"
      eval "${squid_util_var_tosave2}=\"${squid_util_var_tosave}\""
done
])

dnl commit the state changes: deleting the temporary state defined in SQUID_STATE_SAVE
dnl with the same prefix. It's not necessary to specify the extra variables passed
dnl to SQUID_STATE_SAVE again, they will be automatically reclaimed.
AC_DEFUN([SQUID_STATE_COMMIT],[
# commit state, key is $1
unset $1_CFLAGS
unset $1_CXXFLAGS
unset $1_LDFLAGS
unset $1_LIBS
unset $1_CC
unset $1_CXX
for squid_util_var_tosave in $$1_squid_saved_vars
do
      unset ${squid_util_var_tosave}
done
```

```
])

dnl rollback state to the call of SQUID_STATE_SAVE with the same namespace argument.
dnl all temporary state will be cleared, including the custom variables specified
dnl at call time. It's not necessary to explicitly name them, they will be automatically
dnl cleared.
AC_DEFUN([SQUID_STATE_ROLLBACK],[
# rollback state, key is $1
CFLAGS="${$1_CFLAGS}"
CXXFLAGS="${$1_CXXFLAGS}"
LDFLAGS="${$1_LDFLAGS}"
LIBS="${$1_LIBS}"
CC="${$1_CC}"
CXX="${$1_CXX}"
for squid_util_var_tosave in $$1_squid_saved_vars
do
      squid_util_var_tosave2="\$$1_${squid_util_var_tosave}"
      eval "$squid_util_var_tosave=\"${squid_util_var_tosave2}\""
done
SQUID_STATE_COMMIT($1)
])


dnl look for modules in the base-directory supplied as argument.
dnl fill-in the variable pointed-to by the second argument with the
dnl space-separated list of modules
AC_DEFUN([SQUID_LOOK_FOR_MODULES],[
$2=""
for dir in $1/*; do
  module="`basename $dir`"
  if test -d "$dir" && test "$module" != CVS; then
        $2="$$2 $module"
  fi
done
])

dnl remove duplicates out of a list.
dnl dnl argument is the name of a variable to be checked and cleaned up
AC_DEFUN([SQUID_CLEANUP_MODULES_LIST],[
squid_cleanup_tmp_outlist=""
for squid_cleanup_tmp in $$1
do
  squid_cleanup_tmp_dupe=0
  for squid_cleanup_tmp2 in $squid_cleanup_tmp_outlist
```

```
  do
    if test "$squid_cleanup_tmp" = "$squid_cleanup_tmp2"; then
      squid_cleanup_tmp_dupe=1
      break
    fi
  done
  if test $squid_cleanup_tmp_dupe -eq 0; then
    squid_cleanup_tmp_outlist="${squid_cleanup_tmp_outlist} $squid_cleanup_tmp"
  fi
done
$1=$squid_cleanup_tmp_outlist
unset squid_cleanup_tmp_outlist
unset squid_cleanup_tmp_dupe
unset squid_cleanup_tmp2
unset squid_cleanup_tmp
])

dnl check that all the modules supplied as a whitespace-separated list (second
dnl argument) exist as members of the basedir passed as first argument
dnl call AC_MESG_ERROR if any module does not exist. Also sets individual variables
dnl named $2_modulename to value "yes"
dnl e.g. SQUID_CHECK_EXISTING_MODULES([$srcdir/src/fs],[foo_module_candidates])
dnl where $foo_module_candidates is "foo bar gazonk"
dnl checks whether $srcdir/src/fs/{foo,bar,gazonk} exist and are all dirs
dnl AND sets $foo_module_candidates_foo, $foo_module_candidates_bar
dnl and $foo_module_candidates_gazonk to "yes"
AC_DEFUN([SQUID_CHECK_EXISTING_MODULES],[
  for squid_module_check_exist_tmp in $$2
  do
    if test -d $1/$squid_module_check_exist_tmp
    then
      eval "$2_$squid_module_check_exist_tmp='yes'"
      #echo "defining $2_$squid_module_check_exist_tmp"
    else
      AC_MSG_ERROR([$squid_module_check_exist_tmp not found in $1])
    fi
  done
])

dnl lowercases the contents of the variable whose name is passed by argument
AC_DEFUN([SQUID_TOLOWER_VAR_CONTENTS],[
  $1=`echo $$1|tr ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz`
])
```

```
dnl uppercases the contents of the variable whose name is passed by argument
AC_DEFUN([SQUID_TOUPPER_VAR_CONTENTS],[
  $1=`echo $$1|tr abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ`
])

dnl like AC_DEFINE, but it defines the value to 0 or 1 using well-known textual
dnl conventions:
dnl 1: "yes", "true", 1
dnl 0: "no" , "false", 0, ""
dnl aborts with an error for unknown values
AC_DEFUN([SQUID_DEFINE_BOOL],[
squid_tmp_define=""
case "$2" in
  yes|true|1) squid_tmp_define="1" ;;
  no|false|0|"") squid_tmp_define="0" ;;
  *) AC_MSG_ERROR([SQUID_DEFINE[]_BOOL: unrecognized value for $1: '$2']) ;;
esac
ifelse([$#],3,
  [AC_DEFINE_UNQUOTED([$1], [$squid_tmp_define],[$3])],
  [AC_DEFINE_UNQUOTED([$1], [$squid_tmp_define])]
)
unset squid_tmp_define
])

dnl aborts with an error specified as the second argument if the first argument doesn't
dnl contain either "yes" or "no"
AC_DEFUN([SQUID_YESNO],[
if test "$1" != "yes" -a "$1" != "no" ; then
  AC_MSG_ERROR([$2])
fi
])

AC_DEFUN([SQUID_EMBED_BUILD_INFO],[
  AC_ARG_ENABLE([build-info],
    AS_HELP_STRING([--enable-build-info="build info string"],
      [Add an additional string in the output of "squid -v".
        Default is not to add anything. If the string is not specified,
        tries to determine nick and revision number of the current
        bazaar branch]),[
  case "$enableval" in
    no) ${TRUE}
        ;;
    yes)
      if test -d "${srcdir}/.bzr"; then
```

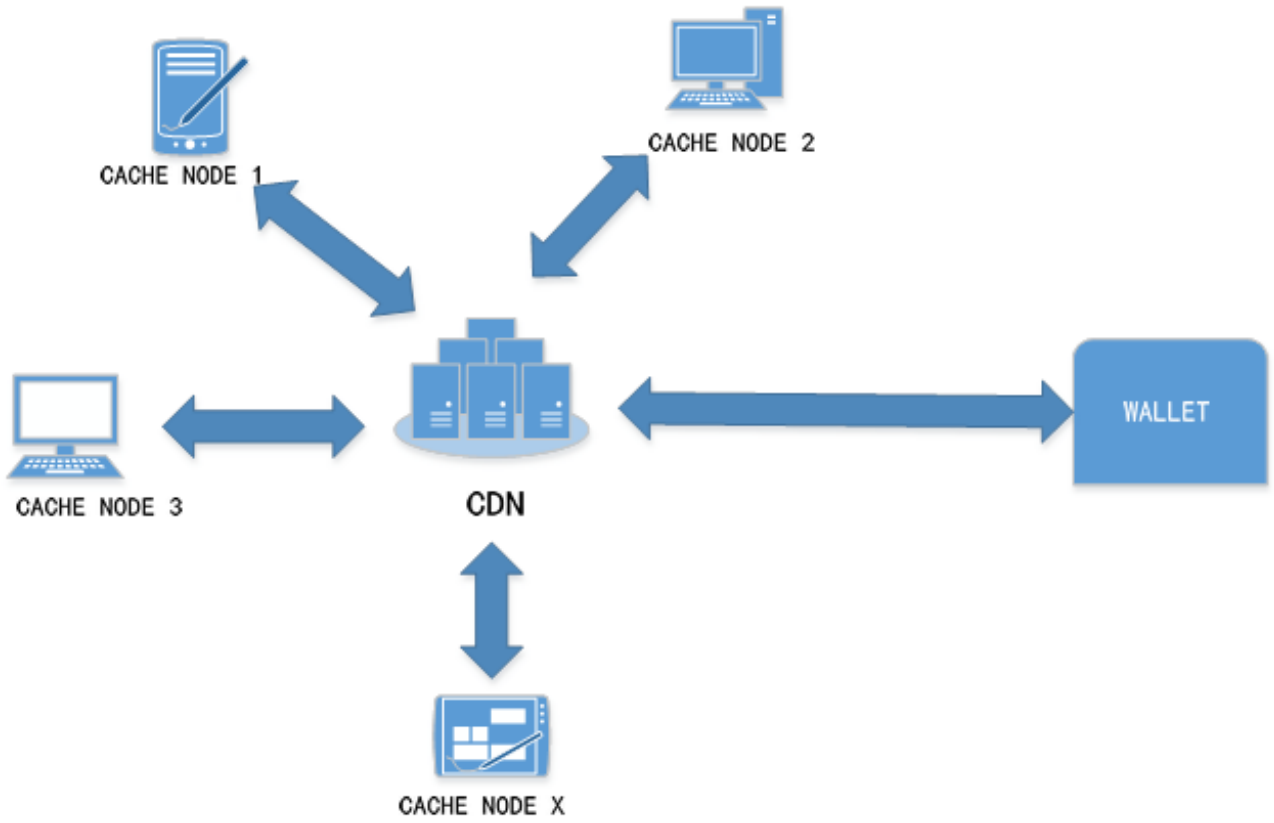
```
            AC_PATH_PROG(BZR,bzr,$FALSE)
            squid_bzr_branch_nick=`cd ${srcdir} && ${BZR} nick 2>/dev/null`
            if test $? -eq 0 -a "x$squid_bzr_branch_nick" != "x"; then
               squid_bzr_branch_revno=`cd ${srcdir} && ${BZR} revno 2>/dev/null | sed 's/\"//g'`
            fi
            if test $? -eq 0 -a "x$squid_bzr_branch_revno" != "x"; then
               sh -c "cd ${srcdir} && ${BZR} diff 2>&1 >/dev/null"
               if test $? -eq 1; then
                     squid_bzr_branch_revno="$squid_bzr_branch_revno+changes"
               fi
            fi
            if test "x$squid_bzr_branch_revno" != "x"; then
               squid_build_info="Built branch: ${squid_bzr_branch_nick}-r${squid_bzr_branch_revno}"
            fi
         fi
         ;;
      *)
         squid_build_info=$enableval
         ;;
   esac
   ])
   AC_DEFINE_UNQUOTED([SQUID_BUILD_INFO],["$squid_build_info"],
      [Squid extended build info field for "squid -v" output])
])

dnl like AC_SEARCH_LIBS, with an extra argument which is
dnl a prefix to the test program
AC_DEFUN([SQUID_SEARCH_LIBS],
[AS_VAR_PUSHDEF([ac_Search], [ac_cv_search_$1])dnl
AC_CACHE_CHECK([for library containing $1], [ac_Search],
[ac_func_search_save_LIBS=$LIBS
AC_LANG_CONFTEST([AC_LANG_PROGRAM([$6], [$1()])])
for ac_lib in '' $2; do
   if test -z "$ac_lib"; then
      ac_res="none required"
   else
      ac_res=-l$ac_lib
      LIBS="-l$ac_lib $5 $ac_func_search_save_LIBS"
   fi
   AC_LINK_IFELSE([], [AS_VAR_SET([ac_Search], [$ac_res])])
   AS_VAR_SET_IF([ac_Search], [break])
done
AS_VAR_SET_IF([ac_Search], , [AS_VAR_SET([ac_Search], [no])])
rm conftest.$ac_ext
```

```
LIBS=$ac_func_search_save_LIBS])
ac_res=AS_VAR_GET([ac_Search])
AS_IF([test "$ac_res" != no],
  [test "$ac_res" = "none required" || LIBS="$ac_res $LIBS"
  $3],
      [$4])
AS_VAR_POPDEF([ac_Search])dnl
])

dnl Check for Cyrus SASL
AC_DEFUN([SQUID_CHECK_SASL],[
  squid_cv_check_sasl="auto"
  AC_CHECK_HEADERS([sasl/sasl.h sasl.h])
  AC_CHECK_LIB(sasl2,sasl_errstring,[LIBSASL="-lsasl2"],[
    AC_CHECK_LIB(sasl,sasl_errstring,[LIBSASL="-lsasl"], [
      squid_cv_check_sasl="no"
    ])
  ])
  case "$squid_host_os" in
    Darwin)
      if test "$ac_cv_lib_sasl2_sasl_errstring" = "yes" ; then
        AC_DEFINE(HAVE_SASL_DARWIN,1,[Define to 1 if Mac Darwin without sasl.h])
        echo "checking for MAC Darwin without sasl.h ... yes"
        squid_cv_check_sasl="yes"
      else
        echo "checking for MAC Darwin without sasl.h ... no"
        squid_cv_check_sasl="no"
      fi
      ;;
  esac
  if test "x$squid_cv_check_sasl" = "xno"; then
    AC_MSG_WARN([Neither SASL nor SASL2 found])
  else
    squid_cv_check_sasl="yes"
  fi
  AC_SUBST(LIBSASL)
])
```

How to employ blockchain smart contract to finish payment



The above picture is the operational framework.

Wallet: digital wallet keeping the Token（BCDN).The wallet can buy the required traffic for website by inputing the website needing acceleration, traffic size, token quantity.

Major CDN: providing CDN service for the website that has bought traffic by getting virtual wallet information

Node: PC, route, TV box and mobilephone, etc can become the cache nodes of CDN and provide distributed traffic for website, and acquire the correspondent token in accordance with the uploaded traffic on the node.

Trading platform: the user can buy and sell BCDN through the third party trading platform.

The above picture is traffic purchasing process.

The user acquires BCDN and keeps it in wallet by crowdfunding or trading website.

The purchase will be finihed as long as the domain name needing acceleration and the quantity of CDN as well as the quantity of BCDN needing payment are input in BlockCDN

The above picture is the exchange process of BCDN.

Major CDN acquires the uploaded traffic of each node and keeps statistics and returns the traffic to wallet which releases the corespondent BCDN according to the given contract rules. The released BCDN can be sold on the the third party trading platform or be used to buy more CDN.

The interactive data that is involved in wallet is recorded in blockchain and can be checked at any time without modification for the purpose of fairness and justice.

After the website owner of CDN is required to release websites needing acceleration and reward on self-service platform BlockCDN, the website data will be deployed on the caching node of the internet and the data will be scattered by SDK to protect the safety of data. When the website is visited by user near some caching node, several nodes near P2P and intelligent CDN scheduling will provide service for the user to finish the visit. The node caching deploying software provides uploaded data (workload) of node for blockchain by API. Blockchain smart contract pays automatically the token BCDN for the node sharer according to website owner's unit price of reward and workload.

Preliminary open source of smart contract is as follows：

https://github.com/BlockChaincdn

```
/*
This file is contract of the BCDN.

The BCDN is free software: you can redistribute it and/or modify
it under the terms of the GNU lesser General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

The BCDN is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.    See the
GNU lesser General Public License for more details.

You should have received a copy of the GNU lesser General Public License
along with the BCDN.    If not, see <http://www.gnu.org/licenses/>.
*/
contract blockcdn {
    mapping (address => uint256) balances;
    address public owner;
    string public name;
    string public symbol;
    uint8 public decimals;
    uint256 public totalSupply;
    uint256 public fundedSupply;
    uint256 public minFundedValue;
    bool public isFunded;
    uint256 closetime;

     /* This generates a public event on the blockchain that will notify clients */
    event Transfer(address indexed from, address indexed to, uint256 value);

    function blockcdn(
        string _tokenName,
        uint8 _decimalUnits,
        string _tokenSymbol,
        uint256 _closetime,
        uint256 _minValue
        ) {
```

```
        owner = msg.sender;                                    // Set owner of the contract
        name = _tokenName;                                     // Set the name for display
purposes
        symbol = _tokenSymbol;                                 // Set the symbol for display
purposes
        decimals = _decimalUnits;                              // Amount of decimals for display
purposes
        closetime = _closetime;                                // Set fund closing time
        minFundedValue = _minValue;                            // Set minimum funing goal
        isFunded = false;                                      // Initialize fund succeed flag
    }

    /*query BCDN balance*/
    function balanceOf( address _owner) constant returns (uint256 balance)
    {
        return balances[_owner];
    }

    /*send ethereum and get BCDN*/
    function buyBlockCDN() returns (bool success){
        if(now > closetime) throw;
        uint256 token = 0;
        if(closetime - 2 weeks > now) {
            token = msg.value;
        }else {
            uint day = (now - (closetime - 2 weeks))/(2 days) + 1;
            token = msg.value;
            while( day > 0) {
                token   =    token * 95 / 100 ;
                day -= 1;
            }
        }

        balances[msg.sender] += token;
        fundedSupply += token;
        balances[owner] = fundedSupply /2;
        totalSupply =   balances[owner] + fundedSupply;
        if(fundedSupply > minFundedValue) {
            isFunded = true;
        }
        Transfer(this, msg.sender, token);
        return true;
    }
```

```
        /*refund 'msg.sender' in the case the Token Sale didn't reach ite minimum
        funding goal*/
        function reFund() returns (bool success) {
            if(now > closetime) throw;
            msg.sender.send(balances[msg.sender]);
            fundedSupply -= balances[msg.sender];
            balances[owner] = fundedSupply /2;
            totalSupply =    balances[owner]    + fundedSupply;
            balances[msg.sender] = 0;
            Transfer(msg.sender, this, balances[msg.sender]);
            return true;

        }

        /* Send coins */
        function transfer(address _to, uint256 _value) returns (bool success) {
            if(now < closetime)    throw;                                    //Closed fund allow transfer
            if (balances[msg.sender] < _value) throw;                       // Check if the sender has enough
            if (balances[_to] + _value < balances[_to]) throw;              // Check for overflows
            balances[msg.sender] -= _value;                                 // Subtract from the sender
            balances[_to] += _value;                                        // Add the same to the recipient
            Transfer(msg.sender, _to, _value);                              // Notify anyone listening that
this transfer took place
            return true;
        }

        /*send reward*/
        function sendRewardBlockCDN(address rewarder, uint256 value) returns (bool success) {
            if(msg.sender != owner) throw;
            if(now <= closetime) throw;
            if( balances[owner] < value) throw;
            balances[rewarder] += value;
            balances[owner] -= value;
            Transfer(owner, rewarder, value);
            return true;

        }

        /*withDraw ethereum when closed fund*/
        function withDrawEth(uint256 value) returns (bool success) {
            if(now <= closetime ) throw;
            if(this.balance < value) throw;
            if(msg.sender != owner) throw;
            msg.sender.send(value);
```

```
                return true;
        }
    }
```

6. Summarize

Being a self-service CDN trade platform, BlockCDN connects the demander and supplier of CDN through blockchain smart contract in a fair, open, transparent way and can effectively reduce the 90% cost of CDN and make fast nodes cover anywhere infinitely with higher network speed and less occupation of network. It can aslo make network users who have idle equipments to share the uploaded traffic and gain benefits without additionl investment. This will be a great revolution in CDN industry.

---

[1] https://en.wikipedia.org/wiki/Content_delivery_network

[2] https://en.wikipedia.org/wiki/Block_chain_(database)

[3] https://en.wikipedia.org/wiki/Sharing_economy

BlockCDN

A blockchain-powered CDN trading platform