Ad Blocking with builds other than Micro

This script, courtesy of several people on the forums, who probably should be named, will enable host-based ad blocking via DNS.

Maybe this script will not work on every device or it might crash your router when there is not enough free ram.

Just put this script into "Startup":

```
#!/bin/sh
logger WAN up script executing
if test -s /tmp/hosts0
then
        rm /tmp/hosts0
fi

logger Downloading http://winhelp2002.mvps.org/hosts.txt
wget -O - http://winhelp2002.mvps.org/hosts.txt | grep 0.0.0.0 |
        sed 's/[[:space:]]*#.*$//g;' |
        grep -v localhost | tr ' ' '\t' |
        tr -s '\t' | tr -d '\015' | sort -u >/tmp/hosts0
grep addn-hosts /tmp/dnsmasq.conf ||
        echo "addn-hosts=/tmp/hosts0" >>/tmp/dnsmasq.conf
logger Restarting dnsmasq
killall dnsmasq
dnsmasq --conf-file=/tmp/dnsmasq.conf
```

...and this into cron:

```
0 12 * * * root /tmp/.rc_startup
```

Enable DNSMasq and Local DNS in Services tab; no Additional DNSMasq options necessary.

Ensure cron is enabled.

You may took alternately this script. Just put this script into "Firewall". Ensure cron is enabled. A cron entry is automatically generated at start-up.

```
# --- COPY THE TEXT BELOW TO DD-WRT / ADMINISTRATION / COMMANDS then click SAVE FIREWALL ---
BH_SCRIPT="/tmp/blocking_hosts.sh"
BH_WHITELIST="/tmp/blocking_hosts.whitelist"
logger "Download blocking hosts file and restart dnsmasq ..."
# Create whitelist. The whitelist entries will be removed from the
# hosts files, i.e. blacklist files.
cat > "$BH_WHITELIST" <<EOF
localhost\\.localdomain
local
invalid
whitelist-example\\.com
```

```
.*\\.whitelist-example\\.com
EOF
# Create download script.
cat > "$BH_SCRIPT" <<EOF
#!/bin/sh
# Function: clean_hosts_file [file ...]
clean_hosts_file() {
  # The sed script cleans up the file.
  # The awk script groups the hosts by ten items.
  sed -e '/^127.0.0.1/b replace;
          /^0.0.0.0/b replace;
          :drop;
            d; b;
          :replace;
            s/^0.0.0.0[[:space:]]*//;
            s/^127.0.0.1[[:space:]]*//;
            s/[[:space:]]*#.*\$//;
            s/[[:space:]]*\$//;
            s/[[:space:]][[:space:]]*/ /;
            /^localhost\$/b drop;
            /^[[:space:]]*\$/b drop;' \$* | \\
  awk 'BEGIN {
         # Read whitelist file.
         n_whitelist = 0
         while ( getline < "$BH_WHITELIST" ) {
           if ( \$0 == "" ) {
             break
           }
           else {
             a_whitelist[++n_whitelist] = \$0
           }
         }
         close("$BH_WHITELIST")
         # Setup record sparator.
         RS=" +"
         c = 0
       }
       {
         for ( n = 1; \$n != ""; n++ ) {
           # Check whitelist.
           whitelist_flag = 0
           for ( w = 1; w <= n_whitelist; w++ ) {
             if ( \$n ~ ( "^" a_whitelist[w] "\$" ) ) {
               whitelist_flag = 1
               break
             }
           }
           if ( whitelist_flag == 0 ) {
             hosts[++c] = \$n
             if ( c == 10 ) {
               s_hosts = "0.0.0.0"
               for ( i = 1; i <= c; i++ ) {
                 s_hosts = s_hosts " " hosts[i]
               }
               print s_hosts
               c = 0
             }
           }
         }
       }
       END {
        if ( c > 0 ) {
```

```
            s_hosts = "0.0.0.0"
            for ( i = 1; i <= c; i++ ) {
              s_hosts = s_hosts = s_hosts " " hosts[i]
            }
            print s_hosts
          }
        }'
}
# Function: wait_for_connection
wait_for_connection() {
  # Wait for an Internet connection.
  # This possibly could take a long time.
  while :; do
    ping -c 1 -w 10 www.freebsd.org > /dev/null 2>&1 && break
    sleep 10
  done
}
# Set lock file.
LOCK_FILE="/tmp/blocking_hosts.lock"
# Check lock file.
if [ ! -f "\$LOCK_FILE" ]; then
  sleep \$((\$\$ % 5 + 5))
  [ -f "\$LOCK_FILE" ] && exit 0
  echo \$\$ > "\$LOCK_FILE"
  # Start downloading files.
  HOSTS_FILE_NUMBER=1
  [ -d "/tmp/blocking_hosts" ] || mkdir "/tmp/blocking_hosts"
  for URL in "http://winhelp2002.mvps.org/hosts.txt" \\
             "http://someonewhocares.org/hosts/zero/hosts" \\
             "http://jansal.googlecode.com/svn/trunk/adblock/hosts" \\
             "http://adblock.gjtech.net/?format=hostfile" \\
             "http://www.hostsfile.org/Downloads/hosts.txt"; do
    HOSTS_FILE="/tmp/blocking_hosts/hosts\`printf '%02d' \$HOSTS_FILE_NUMBER\`"
    logger "Downloading \$URL ..."
    REPEAT=1
    while :; do
      # Wait for internet connection.
      wait_for_connection
      START_TIME=\`date +%s\`
      # Create process to download a hosts file.
      wget -O - "\$URL" 2> /dev/null > "\${HOSTS_FILE}.tmp" &
      WGET_PID=\$!
      WAIT_TIME=\$((\$REPEAT * 10 + 20))
      # Create timeout process.
      ( sleep \$WAIT_TIME; kill -TERM \$WGET_PID ) &
      TIMEOUT_PID=\$!
      wait \$WGET_PID
      CURRENT_RC=\$?
      kill -KILL \$TIMEOUT_PID
      STOP_TIME=\`date +%s\`
      if [ \$CURRENT_RC = 0 ]; then
        clean_hosts_file "\${HOSTS_FILE}.tmp" > "\$HOSTS_FILE"
        rm "\${HOSTS_FILE}.tmp"
        break
      fi
      # In the case of an error: wait the remaining time.
      TIME_SPAN=\$((\$STOP_TIME - \$START_TIME))
      WAIT_TIME=\$((\$WAIT_TIME - \$TIME_SPAN))
      [ \$WAIT_TIME -gt 0 ] && sleep \$WAIT_TIME
      # Increase the number of repeats.
      REPEAT=\$((\$REPEAT + 1))
      [ \$REPEAT = 4 ] && break
```

```
    done
    HOSTS_FILE_NUMBER=\$((\$HOSTS_FILE_NUMBER + 1))
  done
  # Inspect downloaded hosts files.
  ANY_FILE_OK=1
  DNSMASQ_PARAM=""
  for HOSTS_FILE in /tmp/blocking_hosts/hosts[0-9][0-9]; do
    if [ -s "\$HOSTS_FILE" ]; then
      ANY_FILE_OK=0
      DNSMASQ_PARAM=\${DNSMASQ_PARAM:+\$DNSMASQ_PARAM }"--addn-hosts=\$HOSTS_FILE"
    else
      rm "\$HOSTS_FILE"
    fi
  done
  if [ \$ANY_FILE_OK = 0 ]; then
    logger "Restarting dnsmasq with additional hosts file(s) ..."
    killall -TERM dnsmasq
    dnsmasq --conf-file=/tmp/dnsmasq.conf \$DNSMASQ_PARAM &
  fi
  rm "\$LOCK_FILE"
fi
EOF
# Make it executeable.
chmod 755 "$BH_SCRIPT"
# Add crontab entry.
grep -q "$BH_SCRIPT" /tmp/crontab || echo "$(($$ % 60)) 3 * * * root $BH_SCRIPT" >>/tmp/crontab
# Execute script in background.
sh "$BH_SCRIPT" &
```

Its possible to add hosts to a whitelist. Whitelist hosts are regular expressions. You must escape a backslash by a backslash.

You may add other URLs to the list. You may be inspired by Ad-Away. See: Ad-Away-Wiki. Keep in mind: your routers memory limits.

The script works on 32 MB-RAM devices and DIDN'T work on 16 MB-RAM devices! On 16 MB-RAM devices please remove the last download link: http://www.hostsfile.org/Downloads/hosts.txt.

Warning This script is about 5K. Storing it to your firewall script means putting it in your NVRAM. Check to see how much free NVRAM you have. If you don't have enough, you have use jffs or smbfs to remotely mount the script and then call it from the firewall script stored in NVRAM. If you overflow your NVRAM, expect random behavior.

Ad Blocking with Micro build

The Micro build does not support Journaling Flash File System (JFFS and JFFS v. 2). Because of this, the router can be instructed to block ads (though a hosts file) only on a temporary basis. The hosts file can only be written to temporary memory that is lost with each reboot.

As soon as the router is rebooted, loses electrical power, or is restarted with new settings, you must complete the steps below again, in order to again begin ad blocking.

Unfortunately, the Micro build also does not include most functional commands of the Busybox system, including features that would allow the router to automatically download an updated hosts file itself through a

startup script (e.g., the wget command), so you must do it manually. Finally, the Micro build does not include SSH capability, so you must transfer the content of the hosts file by manually pasting it into the router's hosts file through a telnet window, instead of SSH-uploading it to the router.

Steps

1. Download the newest update of a trusted hosts file, such as from MVPS.org at: http://winhelp2002.mvps.org/hosts.txt

2. Open the hosts text file in Notepad, TextEdit, or Notepad++, or another text editor. Highlight all the text in the file and copy it.

3. Go to your router's web interface through your web browser, typically by entering the URL http://192.168.1.1. If you assigned your router a different local IP address, use that instead.

4. In the web interface, click on the Services tab. Enter your DD-WRT router username and password, if prompted.

5. Scroll down to the 'DNSMasq' section. Make sure that DNSMasq bullet and Local DNS bullets are set to 'Enable' that no other DNSMasq options are set.

6. Use Putty or another Telnet client to connect to the router's Telnet interface. (That is, connect to the router's local IP address on port 23 - telnet.)

7. Enter your login and password. The password will be the same as for the web interface. For the username, first try `root`. If that does not work, try the same username as for the web interface.

8. At the DD-WRT command prompt, type `killall DNSMasq` and hit enter.

9. Type `pwd` and hit enter. You should get back "/tmp/root". Type `cd ..` and hit enter.

10. At the next prompt, type `cat > hosts` and hit enter.

11. Paste all the text you copied earlier from the hosts.txt file. (For example, in Putty, bring your mouse cursor to the cursor location at the bottom of the telnet window and click your right mouse button to paste.)

12. After a minute or so of the text pasting/transferring, the screen will stop scrolling. When that happens, finish the file with a Ctrl-D key combination.

13. Type `dnsmasq --conf-file=/tmp/dnsmasq.conf --addn-hosts=/tmp/hosts` and hit enter.

14. You are now finished. Type `exit` and hit enter.

15. Test out your configuration. Use a computer or connected device to use the local commands "ping" or "tracert" on known good sites (like google.com) vs. sites on the hosts list. The known good sites should have much higher ping times than the sites on the hosts list (for example, 12 ms versus less than 1 ms). The "tracert" (traceroute) comamnd should show a much lengthier Internet path to the good site than the bad site (which should only be one or two hops).

16. Again, repeat all these steps if your router goes down or is rebooted for some reason.