chomper Documentation

Release

Aniket Panjwani

Apr 25, 2018

Table of Contents

1 Installation **3**
- 1.1 Debian-based . . . 3
- 1.2 Non Debian-based . . . 3

2 Basic Usage **5**
- 2.1 Types of Rules . . . 5
- 2.2 Configuration File . . . 5
- 2.3 Executing Blocks . . . 6
- 2.4 Overlapping Blocks . . . 6
- 2.5 Resetting a Block . . . 6

3 Advanced Usage **7**
- 3.1 Hardcore Mode . . . 7

4 Development **9**
- 4.1 Overview . . . 9
- 4.2 `chomper.block.enact_block` . . . 9

5 About **11**

6 Module Documentation **13**

Chomper is a Python command-line program which can create blacklists of blocked websites or whitelists of allowed websites for specified periods of time. The program is intended to help people with issues being productive on a computer due to Internet distractions.

Currently, Chomper is only available for GNU Linux, but an OSX port is being developed.

CHAPTER 1

Installation

Chomper can be installed on any Linux distribution. However, the installation has only been fully automated for Debian-based distributions. Automated installation scripts can be found in the Chomper Installers repo.

For each installation method, you will need to be using a user with sudo privileges.

1.1 Debian-based

An automated installation script has been created for Debian-based distributions (e.g. Linux Mint and Ubuntu). You can implement the script with the following command:

```
curl -sL https://raw.githubusercontent.com/aniketpanjwani/chomper_installers/master/
↪debian.sh | bash && source ~/.bashrc
```

Enter your password where prompted. The script will install all UNIX dependencies, install pyenv (a Python version manager) and put it on PATH, install Python 3.6.4 in ~/.pyenv/ subdirectory, clone Chomper into ~/chomper, and append a Chomper executable to your PATH.

1.2 Non Debian-based

There is no automated installation yet for non Debian-based distributions. If you are interested in installing Chomper on your non Debian-based distribution, I suggest looking at the automated installation script for guidance.

If you are on a non-Debian based distribution, want to install Chomper, and are willing to create an installation script for your non-Debian based distribution, please create a new Issue in Github Issues. I would be happy to schedule a time for us to work together to get Chomper up and running on your distribution.

CHAPTER 2

Basic Usage

When using Chomper, you define *rules* of websites to blacklist or whitelist in a YAML file. Blocks are then executed using selected rules for a number of minutes specified by the user.

2.1 Types of Rules

A rule can either be a blacklist or a whitelist

- blacklist: a set of websites which you cannot go to
- whitelist: an exclusive set of websites which you can go to

A rule is composed of addresses, which can be

- domains: e.g. facebook.com
- subdomains: e.g. unix.stackexchange.com
- URLS: e.g. github.com/coleifer/peewee

For example, if unix.stackexchange.com is part of a whitelist rule named *coding*, when *coding* is executed, the user will be able to go to unix.stackexchange.com, but will not be able to go to aviation.stackexchange.com.

2.2 Configuration File

Chomper's rules are configured through a simple YAML file located at ./chomper/data/rules.yaml. The first level specifies the names of the rules. The second level defines whether a rule is a blacklist or whitelist, and the addresses involved in a rule. An example configuration file is found below

```
coding:
  - block_type:
      - whitelist
  - addresses:
    - stackoverflow.com
```

```
    - unix.stackexchange.com
    - python.org
nosocial:
  - block_type:
      - blacklist
  - addresses:
      - facebook.com
      - instagram.com
      - twitter.com
allon:
  - block_type:
      - blacklist
  - addresses:
      - abcdxyz.com
alloff:
  - block_type:
      - whitelist
  - addresses:
      - abcdxyz.com
```

The last two rules, *allon* and *alloff*, are ad-hoc implementations to either allow or block all websites.

2.3 Executing Blocks

Blocks are executed through the syntax `chomper rule_name num_minutes`

For example, with the above YAML file,

- `chomper coding 10` will allow you to only visit those websites under the *coding* rule for 10 minutes
- `chomper nosocial 20` will allow you to visit all websites except those under the *nosocial* rule for 20 minutes

2.4 Overlapping Blocks

When you execute a block, you are locked in to that block. If you try to execute a new block while a block is in progress, you will be prevented from executing the new block.

2.5 Resetting a Block

If you have administrator privileges, you can execute `make reset` from the ./chomper directory. This will restore your Internet access to normal.

However, be wary of the following possibility:

1. You set a block at 11:00 A.M. for 30 minutes.
2. At 11:15 A.M., you reset the block with `make reset`.
3. If, at 11:20 A.M., you try to set a new block, instead, your old block will be enacted (due to the overlapping blocks feature).

CHAPTER 3

Advanced Usage

3.1 Hardcore Mode

If you use Linux and have root privileges, it is impossible to prevent yourself from breaking any sort of Internet block. If you want to use Chomper seriously, I suggest you do the following:

1. Create a new account with administrative privileges. Give this account a very long, complicated password. Write down the password, and store it in some secure, but difficult to access location.
2. While your main account still has root privileges, run `make lock` from the base directory and enter your password where prompted.
3. Remove your main account's administrative privileges, log out, and log back in.

You will now be able to use Chomper using the `chomper` executable the `bin` directory, but you will not be able to edit the code, or kill any processes started by Chomper to block websites.

If you want to keep a back-door which allows you to have delayed access to root privileges, I suggest the nifty delayed-admin tool.

CHAPTER 4

Development

This section describes the Chomper code base for developers interested in contributing to Chomper.

4.1 Overview

Chomper is a wrapper around a Python proxy library called mitmproxy. When the `chomper` executable is called with arguments `rule-name` and `block-time`, the Python interpreter in calls `./chomper/block.py` to enact a block.

`./chomper/block.py` first checks to see if a block is in effect by reading `./data/block_settings.json`.

1. If a block is not in effect, 1. Set a new block (using `chomper.block.enact_block`). 2. Create a new set of block settings (using `chomper.block.create_block_dict`). 3. Schedule a task on root's Crontab to remove the block in `block-time` minutes by calling `./chomper/reset.py`. 4. Provide a message informing users that a new block has been enacted, and tell them when the block will end.
2. If a block is in effect, 1. Reinstate the old block. 2. Provide a message informing users that the old block is still in effect, and tell them when the block will end.

4.2 `chomper.block.enact_block`

This function enacts new blocks. It does the following:

1. Remove any existing rules on iptables' NAT chain.
2. Kill any currently running *mitmdump* or *mitmproxy* processes.
3. Forward all outgoing HTTP/HTTPS traffic from ports 80/443 to port 8080.
4. Start a *mitmdump* process listening on port 8080 as a daemon inside a detached *screen* process.

CHAPTER 5

About

CHAPTER 6

Module Documentation

- genindex
- modindex
- search