CCgen.v2 (Covert Channel generator)

- *... 2020, FM*
- *Nov 2021, FIV*
- *May 2023, KM*

CCgen.v2 is the refactorized version of CCgen.v1 refined and developed by Kaspar Meusburger.

Citation

If you use CCgen for your research, plese refer to our paper in:

Félix Iglesias, Fares Meghdouri, Robert Annessi, Tanja Zseby, "CCgen: Injecting Covert Channels into Network Traffic", Security and Communication Networks, vol. 2022, Article ID 2254959, 11 pages, 2022. https://doi.org/10.1155/2022/2254959

Introduction

CCgen.v2 is a tool for creating, transmitting, and retrieving covert channels into TCP/IP features and packets.

CCgen.v2 has two operation modi: (a) **online**, in which covert channels are generated on-the-fly and transmitted to an aimed destination address, and (b) **offline**, in which covert channels are injected into existing flows of pcap captures. It also shows two operation roles: (1) **injector**, which creates and/or injects a covert channel, and (2) **extractor**, which receives and/or discloses a covert channel.

CCgen.v2 (Link: https://github.com/CN-TU/CCgen.v2) is a new version of CCgen.v1 (Link: https://github.com/CN-TU/py_CCgen) and comes with following improvements and features:

- Improvements:
  - **No** use of **mapping files** anymore -> replaced by mathematic formula.
  - **No** use of **text-based configurations** -> replaced by GUI with strong validation model.
  - Included wrapper functionality which replaces `ccgen_wrapper_*.py`.
- Features:
  - Implementation of **database** to switch between different techniques and modes.
  - Implementation of **wrapper** to inject and extract covert channels automatically
  - Direct **control** for **Spammer VM** without opening any additional terminal or software.
  - Strong **Validation model** stabilizes the software
  - Easy and user-friendly **GUI**.
  - JSON-**interface** to quickly import ordinary and wrapper configurations in JSON format.

PROF

Requirements

To run this application following software and programs are needed:

- **Linux debian-based** operating system (Ubuntu/Debian)
- **Virtualbox** (Version 6.1)

Installation

For installation, run the linux bash script *install.sh* with the command inside a terminal:

-> `bash install.sh`

Run

Open Linux-terminal in the main directory of CCgen.v2. Then do following:

- Execute run script with `sh run.sh` command
- Select if you want to start up online mode too.
- Go to browser and open: `localhost:5000`

CCgen.v2

(1) Dashboard

The *CCgen.v2 Dashboard* has three main functions:

1. **Queue** table:
   Queue of open tasks. After configuring one task or a wrapper, they will land here.
2. **History** table:
   Already finished/done tasks are displayed here.
3. Control of **Spammer Virtual Machine**:
   For online injection, a second device is needed. In this section, one can control the second machine as well as configure it for spamming IP-packets.

(2) Configurator

PROF

The *CCgen.v2 Configurator* is designed to configure injection and extraction tasks in all four modes. The modes are:

- (a1) online injector
- (a2) online extractor
- (b1) offline injector
- (b2) offline extractor.

The extractor functionality is designed for proving if injection has worked correctly.

In general, one configuration can be separated in three sub-categories:

- **General**
  In this section general data of one configuration are saved. It consists the name (for saving in db), input/output file paths and message (which will be injected). The source of message can either be a link to a file on the server machine, or it can be entered directly into text box.

- **Filters**
  In this section the connection/conversation is defined. Therefore, filters for source/destination IP (ipv4 & ipv6), source/destination port (tcp/udp) and the ip protocol number can be configured. Depending on network mode (online/offline), the filteres are appended to a pre-captured .pcap file (offline) or to the network interface (online).

- **Channel & Mapping**
  In this section the technique of the covert channel is defined. The CCgen.v2 is built, that every technique has a own script. This script is referenced in this part of the configuration. Additionally it is mandatory to configure the number of bits, the layer as well as parameters and value mappings.
  Parameters are used to customizeably define parameters, which are referenced in the given technique file (python script).
  Value mappings are specified, if one bit value is not converted directly. (E.g. Normally "b'01" -> 1, but with specified value mapping the value could be mapped like: "b'01" -> 24)

- **Config wrapper**
  The main purpose of the CCgen-wrapper is to allow the automatic injection of multiple covert channels in the same pcap. It has three different parts:

  1. Searching for matching flows and creating corresponding ccGen.v2 configurations.
  2. Injecting the covert channels.
  3. Extracting the covert channels to evaluate the previous injection.

This mode can be used by creating a new configuration for **offline injection** by clicking the **Add to wrapper** button after successful validation. To validate the wrapper config, click Validate button in top right corner.

For more information, read the README.md file in `~/CCgen.v2/ccgen/wrapper/README.md`.

(3) Settings

PROF

The *CCgen.v2 Settings* is intended to pre-configure input fields of the *CCgen.v2 Configurator*. This should ease the use of the configurator.
Each mode (a1,a2,b1,b2) of the configurator can have its own presets. They are saved to database after validation. If "create new config" is selected in the *CCgen.v2 Configurator*, the preset matching the network and direction select, is preloaded and the input fields are set with its values.

(4) Interface

The *CCgen.v2 Interface* is a JSON interface to quickly import CCgen.v2 configurations in JSON format. Single configuration can be send either to the *CCgen.v2 Configurator* where the configuration can be changed or they can be send directly to the CCgen.v2 core module. Wrapper configurations can only be send to the CCgen.v2 core module, where they are processed.

Examples

(a1) Online injection

1. Go to Dashboard `localhost:5000/`
2. Start Spammer Virtual Machine by clicking `Start` button
3. Check if `Listener` is `running`
4. Check if `Spammer` is `running`
5. Go to Configurator `localhost:5000/configurator`
6. Select configuration `ipfalgs_v2s`
7. Select network `online`
8. Select direction `inject`
9. Adjust source ip address to **your** pc ip address -> find with terminal command `ifconfig`
10. Adjust destination ip address to ip address of **SpammerVM** -> find on CCgen.v2 Dashboard
11. Set `IP tables queue` (default=0)
12. Click button `Validate`
13. Click button `Save config & queue simulation`

(optional)

13. Start `wireshark`
14. Capture traffic during generation
15. Save capture as .pcap file
16. Check injection with `offline extractor` (b2)

(a2) Online extraction

This mode was not tested with experiments.

(b1) Offline injection

1. Go to Configurator `localhost:5000/configurator`
2. Select configuration `ipfalgs_v2s`
3. Select network `offline`
4. Select direction `inject`
5. Adjust input to `~/CCgen.v2/pcaps/empty.pcap`
6. Adjust output, preferred to `~/CCgen.v2/pcaps/ipflags_v2s.pcap`
7. Adjust source ip address to `200.60.153.149`
8. Adjust destination ip address to `15.161.162.34`
9. Validate
10. Save config & queue simulation

PROF

(b2) Offline extraction

Follow these steps after injecting config `ipflags_v2s` as described in (b1).

1. Go to Configurator `localhost:5000/configurator`
2. Select configuration `ipfalgs_v2s`
3. Select network `offline`
4. Select direction `extract`
5. Adjust input to `~/CCgen.v2/pcaps/ipflags_v2s.pcap`
6. Adjust output, preferably to `~/CCgen.v2/out/ipflags_v2s.txt`

7. Adjust source ip address to `200.60.153.149`
8. Adjust destination ip address to `15.161.162.34`
9. Click button `Validate`
10. Click button `Save config & queue simulation`