D4 Project

Open and collaborative network monitoring

`https://www.d4-project.org/`

2019/12/06

Team CIRCL

Problem statement

- CSIRTs (or private organisations) build their **own honeypot, honeynet or blackhole monitoring network**
- Designing, managing and operating such infrastructure is a tedious and resource intensive task
- **Automatic sharing** between monitoring networks from different organisations is missing
- Sensors and processing are often seen as blackbox or difficult to audit

Objective

- Based on our experience with MISP[1] where sharing played an important role, we transpose the model in D4 project
- Keeping the protocol and code base **simple and minimal**
- Allowing every organisation to **control and audit their own sensor network**
- Extending D4 or **encapsulating legacy monitoring protocols** must be as simple as possible
- Ensuring that the sensor server has **no control on the sensor** (unidirectional streaming)
- Don't force users to use dedicated sensors and allow **flexibility of sensor support** (software, hardware, virtual)

[1] https://github.com/MISP/MISP

(short) History

- D4 Project (co-funded under INEA CEF EU program) started - **1st November 2018**
- D4 encapsulation protocol version 1 published - **1st December 2018**
- v0.1 release of the D4 core[2] including a server and simple D4 C client - **21st January 2019**
- First version of a golang D4 client[3] running on ARM, MIPS, PPC and x86 - **January 2019**
- First Analyzers - **Spring 2019**
- Client Generator - **Summer 2019**

[2] https://www.github.com/D4-project/d4-core

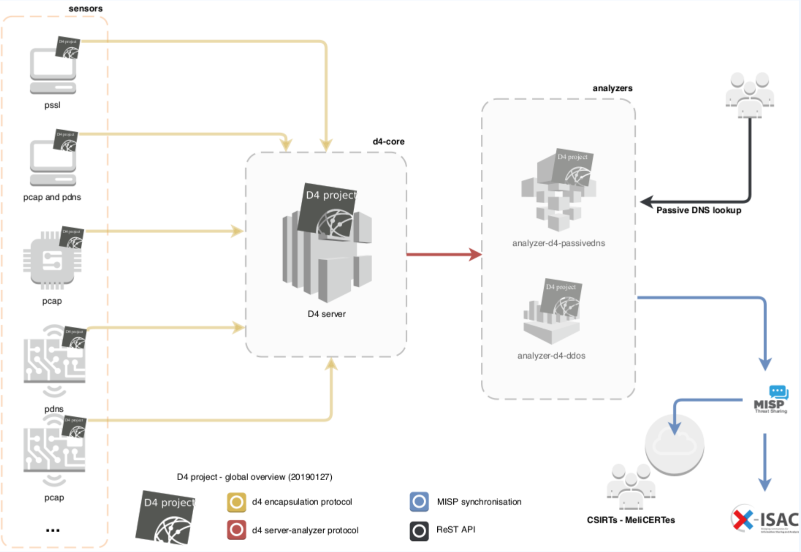
[3] https://www.github.com/D4-project/d4-goclient/

(short) History

Release	Date
AIL-framework-v1.5	Apr. 26, 2019
...	
AIL-framework-v2.1	Aug. 14, 2019
analyzer-d4-balboa-v0.1	Aug. 19, 2019
analyzer-d4-passivedns-v0.1	Apr. 5, 2019
analyzer-d4-passivessl-0.1	Apr. 25, 2019
analyzer-d4-pibs-v0.1	Apr. 8, 2019
BGP-Ranking-1.0	Apr. 25, 2019
BGP-Ranking-1.1	Aug. 19, 2019
d4-core-v0.1	Jan. 25, 2019
d4-core-v0.2	Feb. 14, 2019
d4-core-v0.3	Apr. 8, 2019
d4-goclient-v0.1	Feb. 14, 2019
d4-goclient-v0.2	Apr. 8, 2019
d4-sensor-generator-v0.1	Aug. 22, 2019
d4-server-packer-0.1	Apr. 25, 2019
IPASN-History-1.0	Apr. 25, 2019
IPASN-History-1.1	Aug. 19, 2019
sensor-d4-tls-fingerprinting-0.1	Apr. 25, 2019

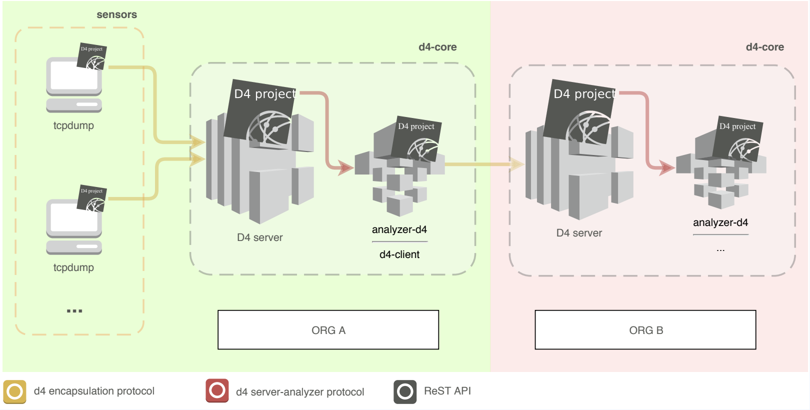
see `https://github.com/D4-Project`

D4 Overview



sensors
pssl
pcap and pdns
pcap
pdns
pcap
...
d4-core
D4 project
D4 server
analyzers
analyzer-d4-passivedns
analyzer-d4-ddos
Passive DNS lookup
MISP
Threat Sharing
CSIRTs - MeliCERTes
X-ISAC
D4 project - global overview (20190127)
D4 project
d4 encapsulation protocol
d4 server-analyzer protocol
MISP synchronisation
ReST API

D4 Overview - Connecting Sensor Networks



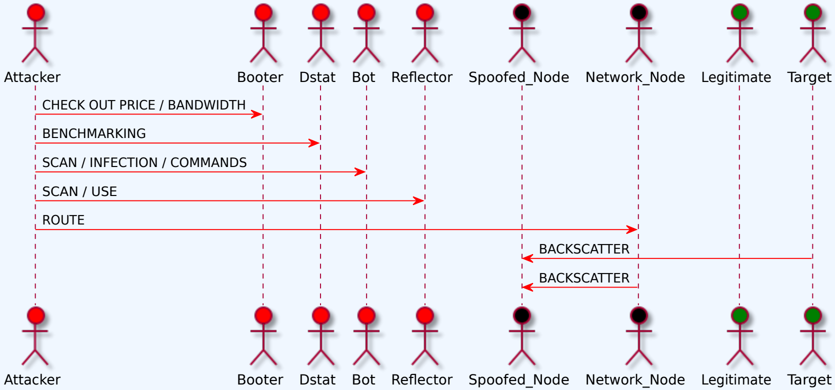
https://d4-project.org/2019/06/17/sharing-between-D4-sensors.html

What to do with it

- Passive DNS collection
- Passive SSL collection
- AIL collection
- Correlations, CTI
- DDoS Detection

D4 Overview: DDoS

DDoS eavesdropping locations.



https://d4-project.org/2019/08/29/state-of-the-art-DDoS.html

Roadmap - output

CIRCL hosts a server instance for organisations willing to contribute to a public dataset without running their own D4 server:

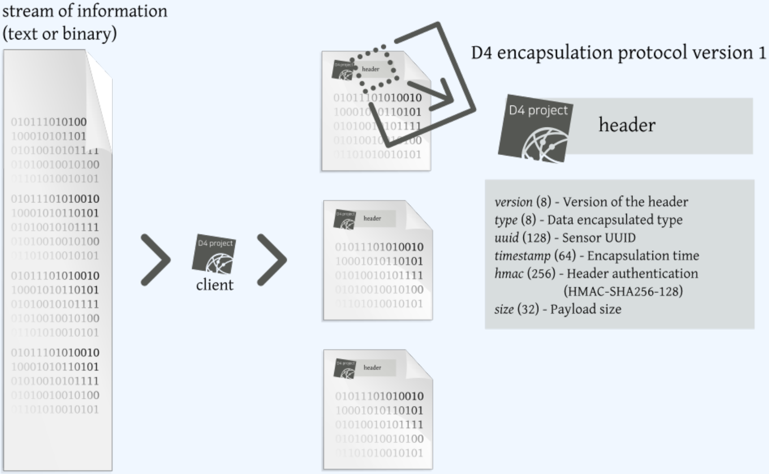
- ✓ Blackhole DDoS
- ✓ Passive DNS
- ✓ Passive SSL
- Gene[4] / WHIDS[5] (sysmon)
- Maltrail[6]
- BGP mapping
- egress filtering mapping
- Radio-Spectrum monitoring: 802.11, BLE, ~~GSM~~, etc.

[4] https://github.com/0xrawsec/gene

[5] https://github.com/0xrawsec/whids

[6] https://github.com/stamparm/maltrail

D4 encapsulation protocol



stream of information
(text or binary)

D4 project
client

D4 encapsulation protocol version 1

D4 project
header

version (8) - Version of the header
type (8) - Data encapsulated type
uuid (128) - Sensor UUID
timestamp (64) - Encapsulation time
hmac (256) - Header authentication
(HMAC-SHA256-128)
size (32) - Payload size

https://www.d4-project.org

D4 Header

Name	bit size	Description
version	uint 8	Version of the header
type	uint 8	Data encapsulated type
uuid	uint 128	Sensor UUID
timestamp	uint 64	Encapsulation time
hmac	uint 256	Authentication header (HMAC-SHA-256-128)
size	uint 32	Payload size

D4 Header

Type	Description
0	Reserved
1	pcap (libpcap 2.4)
2	meta header (JSON)
3	generic log line
4	dnscap output
5	pcapng (diagnostic)
6	generic NDJSON or JSON Lines
7	generic YAF (Yet Another Flowmeter)
8	passivedns CSV stream
254	type defined by meta header (type 2)

D4 meta header

D4 header includes an easy way to **extend the protocol** (via type 2) without altering the format. Within a D4 session, the initial D4 packet(s) type 2 defines the custom headers and then the following packets with type 254 is the custom data encapsulated.

```
{
  "type": "ja3-jl",
  "encoding": "utf-8",
  "tags": [
    "tlp:white"
  ],
  "misp:org": "5b642239-4db4-4580-adf4-4ebd950d210f"
}
```

D4 server

- D4 core server[7] is a complete server to handle clients (sensors) including the decapsulation of the D4 protocol, control of sensor registrations, management of decoding protocols and dispatching to adequate decoders/analysers.
- D4 server is written in Python 3.6 and runs on standard GNU/Linux distribution.

[7] https://github.com/D4-project/d4-core

D4 server handling

D4 server reconstructs the encapsulated stream from the D4 sensor and saves it in a Redis stream.

- Support TLS connection
- Unpack D4 header
- Verify client secret key (HMAC)
- check blocklist
- Filter by types (Only accept one connection by type-UUID - except: type 254)
- Discard incorrect data
- Save data in a Redis Stream (unique for each session)

D4 server - worker handler

After the stream is processed depending of the type using dedicated worker.

- Worker Manager (one by type)
  - Check if a new session is created and valid data are saved in a Redis stream
  - Launch a new Worker for each session
- Worker
  - Get data from a stream
  - Reconstruct data
  - Save data on disk (with file rotation)
  - Save data in Redis. Create a queue for D4 Analyzer(s)

D4 server - type 254 worker handler

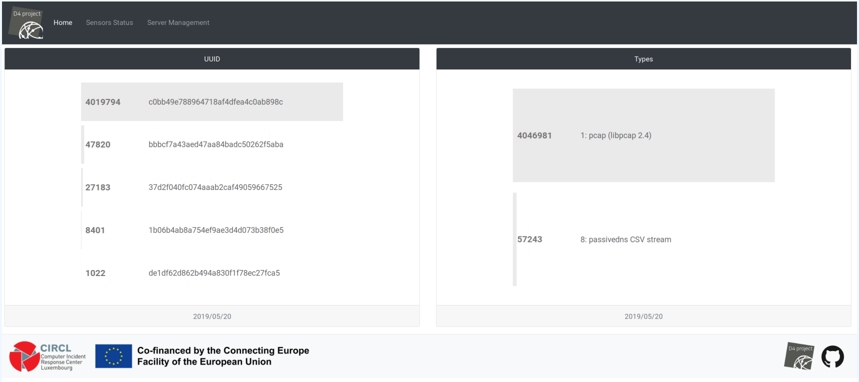
- Worker custom type (called Worker 2)
  - Get type 2 data from a stream
  - Reconstruct Json
  - Extract extended type name
  - Use default type or special extended handler
  - Save Json on disk
  - Get type 254 data from a stream
  - Reconstruct type 254
  - Save data in Redis. Create a queue for D4 Analyzer(s)

D4 server - management interface

The D4 server provides a **web interface** to manage D4 sensors, sessions and analyzer.

- Get Sensors status, errors and statistics
- Get all connected sensors
- Manage Sensors (stream size limit, secret key, ...)
- Manage Accepted types
- UUID/IP blocklist
- Create Analyzer Queues

D4 SERVER - MAIN INTERFACE



UUID

4019794	c0bb49e788964718af4dfea4c0ab898c
47820	bbbcf7a43aed47aa84badc50262f5aba
27183	37d2f040fc074aaab2caf49059667525
8401	1b06b4ab8a754ef9ae3d4d073b38f0e5
1022	de1df62d862b494a830f1f78ec27fca5

2019/05/20

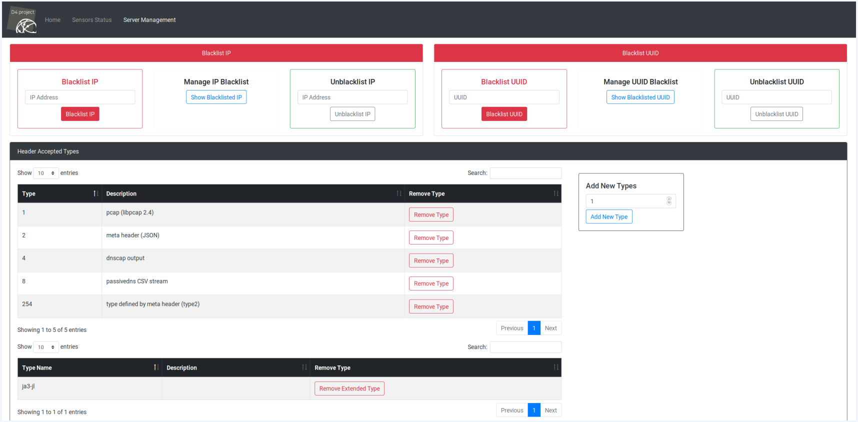
Types

4046981	1: pcap (libpcap 2.4)
57243	8: passivedns CSV stream

2019/05/20

Co-financed by the Connecting Europe Facility of the European Union

D4 SERVER - SERVER MANAGEMENT



Blacklist IP

Blacklist IP

IP Address

Blacklist IP

Manage IP Blacklist

Show Blacklisted IP

Unblacklist IP

IP Address

Unblacklist IP

Blacklist UUID

Blacklist UUID

UUID

Blacklist UUID

Manage UUID Blacklist

Show Blacklisted UUID

Unblacklist UUID

UUID

Unblacklist UUID

Header Accepted Types

Show 10 entries Search:

Type	Description	Remove Type
1	pcap (libpcap 2.4)	Remove Type
2	meta header (JSON)	Remove Type
4	dnscap output	Remove Type
8	passivedns CSV stream	Remove Type
254	type defined by meta header (type2)	Remove Type

Showing 1 to 5 of 5 entries Previous 1 Next

Show 10 entries Search:

Type Name	Description	Remove Type
ja3-jl		Remove Extended Type

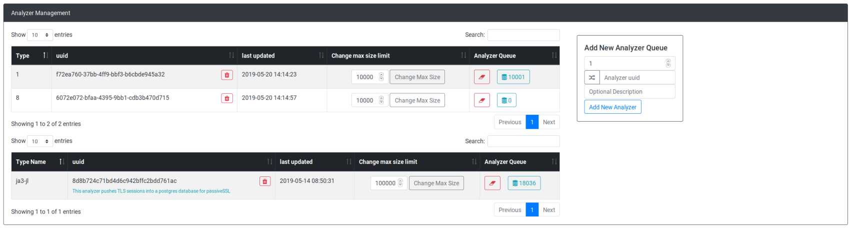
Showing 1 to 1 of 1 entries Previous 1 Next

Add New Types

1

Add New Type

D4 SERVER - SERVER MANAGEMENT



Analyzer Management

Show 10 entries

Search:

Type	uuid	last updated	Change max size limit	Analyzer Queue
1	f72ea760-37bb-4ff9-bbf3-b6cbde945a32	2019-05-20 14:14:23	10000 Change Max Size	10001
8	6072e072-bfaa-4395-9bb1-cdb3b470d715	2019-05-20 14:14:57	10000 Change Max Size	0

Showing 1 to 2 of 2 entries

Show 10 entries

Search:

Type Name	uuid	last updated	Change max size limit	Analyzer Queue
ja3-jl	8d8b724c71bd4d6c942bffc2bdd761ac This analyzer pushes TLS sessions into a postgres database for passiveSSL	2019-05-14 08:50:31	100000 Change Max Size	18036

Showing 1 to 1 of 1 entries

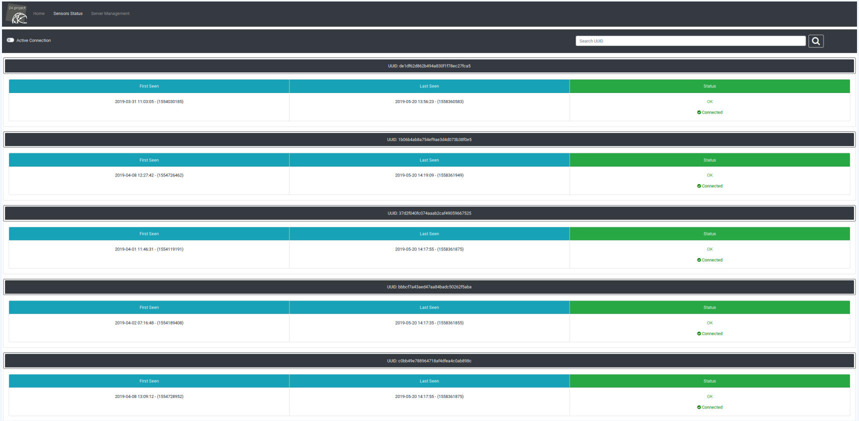
Add New Analyzer Queue

1

Analyzer uuid

Optional Description

D4 server - sensor overview



UUID: de1df62d862b494a830f1f78ec27fca5

First Seen	Last Seen	Status
2019-03-31 11:03:05 - (1554030185)	2019-05-20 13:56:23 - (1558360583)	OK ✓ Connected

UUID: 1b06b4ab8a754ef9ae3d4d073b38f0e5

First Seen	Last Seen	Status
2019-04-08 12:27:42 - (1554726462)	2019-05-20 14:19:09 - (1558361949)	OK ✓ Connected

UUID: 37d2f040fc074aaab2caf49059667525

First Seen	Last Seen	Status
2019-04-01 11:46:31 - (1554119191)	2019-05-20 14:17:55 - (1558361875)	OK ✓ Connected

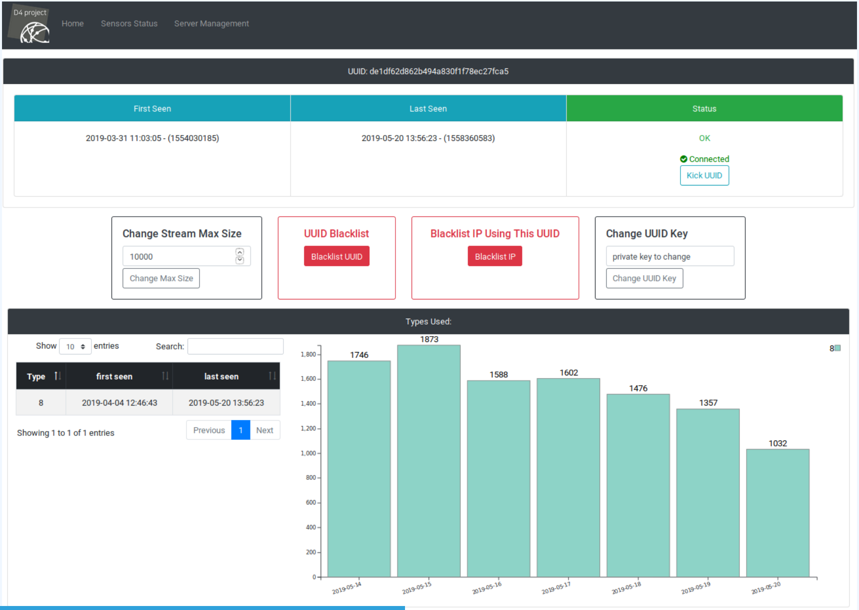
UUID: bbbcf7a43aed47aa84badc50262f5aba

First Seen	Last Seen	Status
2019-04-02 07:16:48 - (1554189408)	2019-05-20 14:17:35 - (1558361855)	OK ✓ Connected

UUID: c0bb49e788964718af4dfea4c0ab898c

First Seen	Last Seen	Status
2019-04-08 13:09:12 - (1554728952)	2019-05-20 14:17:55 - (1558361875)	OK ✓ Connected

D4 SERVER - SENSOR MANAGEMENT



UUID: de1df62d862b494a830f1f78ec27fca5

First Seen	Last Seen	Status
2019-03-31 11:03:05 - (1554030185)	2019-05-20 13:56:23 - (1558360583)	OK ✓ Connected Kick UUID

Change Stream Max Size

10000

Change Max Size

UUID Blacklist

Blacklist UUID

Blacklist IP Using This UUID

Blacklist IP

Change UUID Key

private key to change

Change UUID Key

Types Used:

Show 10 entries Search:

Type	first seen	last seen
8	2019-04-04 12:46:43	2019-05-20 13:56:23

Showing 1 to 1 of 1 entries

Previous 1 Next

Example use-case: migrating a legacy network capture model into a D4 network sensor

Remote network capture

CIRCL operated honeybot for multiple years using a simple model of remote network capture.

Definition (Principle)

- KISS (Keep it simple stupid) - Unix-like
- Linux & OpenBSD operating systems

Sensor

```
tcpdump -l -s 65535 -n -i vr0 -w - ’(␣not␣port␣
   $PORT␣and␣not␣host␣$HOST␣)’ | socat -
   OPENSSL-CONNECT:$COLLECTOR:$PORT,cert=/etc/
   openssl/client.pem,cafile=/etc/openssl/ca.crt,
   verify=1
```

Remote network capture

Limitations

- Scalability → one port per client
- Identification and registration of the client
- Integrity of the data

Encapsulating streams in D4

- Inspired by the unix command `tee`
- Read from standard input
- Add the d4 header
- Write it on standard output

Using D4 native client

```
tcpdump -n -s0 -w - | ./d4 -c ./conf | socat -
   OPENSSL-CONNECT:$D4-SERVER-IP-ADDRESS:$PORT,
   verify=1
```

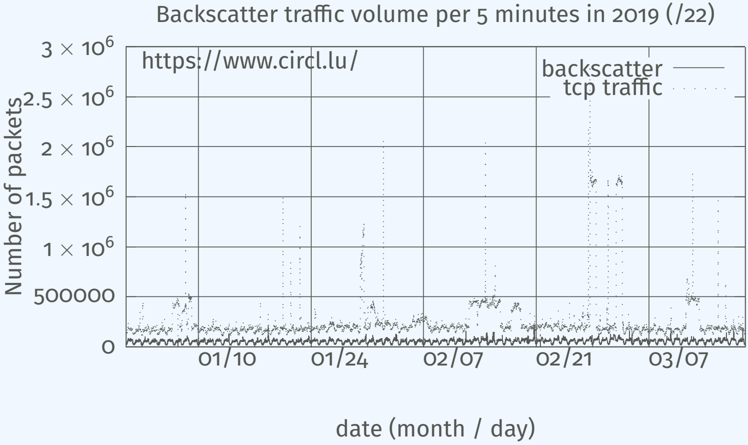
Configuration directory

Parameter	Explanation
type	see D4 Header slide
source	standard input
key	HMAC key
uuid	Identifier of the sensor
version	version of the sensor
destination	standard output
snaplen	length of data being read & written

A distributed Network telescope to observe DDoS attacks

Motivation

DDoS Attacks produce an observable side-effect:



What can be derived from backscatter traffic?

- External point of view on ongoing Denial of Service attacks:
  - **Confirm** if there is a DDoS attack
  - **Recover** time line of attacked targets
  - **Confirm** which services (DNS, webserver, . . .)
  - **Observe** Infrastructure changes
- **Assess the state of an infrastructure under denial of service attack**
  - **Detect** failure/addition of intermediate network equipments, firewalls, proxy servers etc
  - **Detect** DDoS mitigation devices
- **Create** models of DoS/DDoS attacks

D4 in this setting

D4 - for data collection and processing:

- **provide** various points of observation in non contiguous address space,
- **aggregate** and **mix** backscatter traffic collected from D4 sensors,
- **perform** analysis on big amount of data.

D4 - from a end-user perspective:

- **provide** backscatter analysis results,
- **provide** daily updates,
- **provide** additional relevant (or pivotal) information (DNS, BGP, etc.),
- **provide** an API and search capabilities.

First release

- ✓ analyzer-d4-pibs[8], an analyzer for a D4 network sensor:
  - **processes** data produced by D4 sensors (pcaps),
  - **displays** potential backscatter traffic on standard output,
  - **focuses** on TCP SYN flood in this first release.
- ■ analyzer-d4-ipa[9],
  - **processes** data produced by D4 sensors (pcaps),
  - **analyze** ICMP packets,

[8] https://github.com/D4-project/analyzer-d4-pibs

[9] https://github.com/D4-project/analyzer-d4-ipa

Passive DNS

Problem statement

- CIRCL (and other CSIRTs) have their own passive DNS[10] collection mechanisms
- Current **collection models** are affected with DoH[11] and centralised DNS services
- DNS answers collection is a tedious process
- **Sharing Passive DNS stream** between organisation is challenging due to privacy

[10] https://www.circl.lu/services/passive-dns/

[11] DNS over HTTPS

Potential Strategy

- Improve **Passive DNS collection diversity** by being closer to the source and limit impact of DoH (e.g. at the OS resolver level)
- Increasing diversity and **mixing models** before sharing/storing Passive DNS records
- Simplify process and tools to install for **Passive DNS collection by relying on D4 sensors** instead of custom mechanisms
- Provide a distributed infrastructure for mixing streams and filtering out the sharing to the validated partners

First release

- ✓ analyzer-d4-passivedns[12], an analyzer for a D4 network sensor:
  - ▶ **processes** data produced by D4 sensors (in passivedns CSV format[13]),
  - ▶ **ingests** these into a **Passive DNS server** which can be queried later to search for the Passive DNS records,
  - ▶ **provides** a lookup server (using on redis-compatible backend) that is a Passive DNS REST server compliant to the Common Output Format[14].

[12] https://github.com/D4-project/analyzer-d4-passivedns

[13] https://github.com/gamelinux/passivedns

[14] https://tools.ietf.org/html/draft-dulaunoy-dnsop-passive-dns-cof-04

Common Output Format

- **Consistent naming of fields across Passive DNS software** based on the most common Passive DNS implementations
- Minimal set of fields to be supported
- Minimal set of optional fields to be supported
- Way to add "additional" fields via a simple registry mechanism (IANA-like)
- Simple and easily parsable format
- A gentle reminder regarding privacy aspects of Passive DNS

Sample output www.terena.org

```
{"count": 868, "time_first": 1298398002, "rrtype": "A", "rrname": "www.terena.org", "rdata": "192.87.30.6", "time_last": 1383124252}
{"count": 89, "time_first": 1383729690, "rrtype": "CNAME", "rrname": "www.terena.org", "rdata": "godzilla.terena.org", "time_last": 1391517643}
{"count": 110, "time_first": 1298398002, "rrtype": "AAAA", "rrname": "www.terena.org", "rdata": "2001:610:148:dead::6", "time_last": 136670845}
```

Mandatory fields

- **rrname** : name of the queried resource records
  - JSON String
- **rrtype** : resource record type
  - JSON String (interpreted type of resource type if known)
- **rdata** : resource records of the query(ied) resource(s)
  - JSON String or an array of string if more than one unique triple
- **time_first** : first time that the resource record triple (rrname, rrtype, rdata) was seen
- **time_last** : last time that the resource record triple (rrname, rrtype, rdata) was seen
  - JSON Number (epoch value) UTC TZ

Optional fields

- **count** : how many authoritative DNS answers were received by the Passive DNS collector
  - JSON Number
- **bailiwick** : closest enclosing zone delegated to a nameserver served in the zone of the resource records
  - JSON String

Additionals fields

- **sensor_id** : Passive DNS sensor information
  - JSON String
- **zone_time_first** : specific first/last time seen when imported from a master file
- **zone_time_last**
  - JSON Number
- Additional fields can be requested via `https://github.com/adulau/pdns-qof/wiki/Additional-Fields`

Passive SSL revamping

Objectives - TLS Fingerprinting

Keep a log of links between:

- x509 certificates,
- ports,
- IP address,
- client (ja3),
- server (ja3s),

  "JA3 is a method for creating SSL/TLS client fingerprints that should be easy to produce on any platform and can be easily shared for threat intelligence."[15]

Pivot on additional data points during Incident Response

[15] https://github.com/salesforce/ja3

Objectives - Mind your Ps and Qs

Collect and **store** x509 certificates and TLS sessions:

- Public keys type and size,
- moduli and exponents,
- curves parameters.

Detect anti patterns in crypto:

- Shared Public Keys,
- Moduli that share one prime factor,
- Moduli that share both prime factor,
- Small factors,
- Nonces reuse / common prefix or suffix, etc.

First release

- ✓ sensor-d4-tls-fingerprinting [16]: **Extracts** and **fingerprints** certificates, and **computes** TLSH fuzzy hash.
- ✓ analyzer-d4-passivessl [17]: **Stores** Certificates / PK details in a PostgreSQL DB.
- ■ snake-oil-crypto [18]: **Runs** weak crypto attacks against the dataset.
- ■ lookup-d4-passivessl [19]: **Exposes** the DB through a public REST API.

[16] github.com/D4-project/sensor-d4-tls-fingerprinting

[17] github.com/D4-project/analyzer-d4-passivessl

[18] github.com/D4-project/snake-oil-crypto

[19] github.com/D4-project/lookup-d4-passivessl

Future

- **Sensitive information sanitization** by specialized analyzers
- **Previewing datasets** collected in D4 sensor network and providing **open data stream** (if contributor agrees to share under specific conditions)
- **Leverage MISP sharing communities** to augment Threat Intelligence, and provide accurate metrology.

Use it

- **Create** sensors easily with the generator [20],
- **Manage** your own sensors and servers, **find** shameful bugs and **fill** in github issues
- Even better, **send** Pull Requests!
- **Share** data to public servers to improve the datasets (and detection, response, etc.)
- **Feed** your MISP instances with D4's findings - **Share** yours
- **Leech** data, **write** your own analyzers, **do** research

[20] https://github.com/d4-project/d4-sensor-generator

Get in touch if you want to join the project, host a sensor or contribute

- Collaboration can include research partnership, sharing of collected streams or improving the software.
- Contact: info@circl.lu
- `https://github.com/D4-Project`
- `https://twitter.com/d4_project`
- `https://d4-project.org`
  - Passive DNS tutorial
  - Data sharing tutorial