CERT.LV

Information Technologies Security Incident Response Institution

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

Pastelyzer

The paste analyzer

Jānis Džeriņš

TF-CSIRT & FIRST Symposium, Malaga, Jan 30, 2020

CERT.LV

Outline

Paste sites and pastes

- Sites to share text documents
  - pastebin.com
  - gist.github.com
  - Many other
- Used to discuss something online
  - Code fragments
  - Error messages (troubleshooting)
- In most cases pastes are anonymous and public
- Security and privacy related information shared often

Enter pastelyzer (1)

- Work in progress!
- Analyzes text documents
  - Not limited to pastes

- Detects privacy- and security-related information
  - Credentials
  - Bank card numbers
  - IP addresses
  - Domains
  - etc.
- No fuss approach
  - Easy installation
  - No babysitting

Enter *pastelyzer* (1)

- Work in progress!
- Analyzes text documents
  - Not limited to pastes
  - Not limited to text documents
- Detects privacy- and security-related information
  - Credentials
  - Bank card numbers
  - IP addresses
  - Domains
  - etc.
- No fuss approach
  - Easy installation
  - No babysitting

Enter *pastelyzer* (2)

- Recognizes binary data encoded in text document (e.g., Base64)
  - Often found in malicious scripts
- Analyzes decoded content (assumed binary)
  - Has basic support for encoding detection
    - Usually ISO-8859-X or UTF-8
    - UTF-16 (little-endian) used on Windows
  - Or maybe the decoded content is a compressed document?
    - Currently supported compression methods: `gzip`, `zlib`, `deflate`

Enter *pastelyzer* (2)

- Recognizes binary data encoded in text document (e.g., Base64)
  - Often found in malicious scripts
  - Since last year AIL can also do this
- Analyzes decoded content (assumed binary)
  - Has basic support for encoding detection
    - Usually ISO-8859-X or UTF-8
    - UTF-16 (little-endian) used on Windows
  - Or maybe the decoded content is a compressed document?
    - Currently supported compression methods: `gzip`, `zlib`, `deflate`

CERT.LV

Outline

Overview

- Pastes provided by CIRCL.LU feed
    - Other ingestion methods planned (e.g. API submission)
- Documents stored in SQL database
- Special actions can be added through configuration
- Command-line interface

The database

- It's SQL!
  - Easy to query and build additional tools on top
- Information stored
  - Document origin and time
  - Document content
  - Summary of discoveries
- DB size: `~200GB` in 3 years

The database (cont.)

- Example: find last 3 documents with more than 50K credentials

```sql
SELECT content_id, summary
  FROM analysis
 WHERE summary #> '{CREDENTIAL, UNIQUE}' > '50000'::jsonb
 ORDER BY content_id DESC
 LIMIT 3;
```

The database (cont.)

```
40732971 {"EMAIL":               {"UNIQUE": 6},
          "DOMAIN":              {"UNIQUE": 4, "DUPLICATE": 41},
          "CREDENTIAL":          {"UNIQUE": 100459},
          "RESOLVED-IP-ADDRESS": {"UNIQUE": 8}}
39969813 {"EMAIL":               {"UNIQUE": 76},
          "DOMAIN":              {"UNIQUE": 1},
          "CREDENTIAL":          {"UNIQUE": 98928},
          "BANK-CARD-NUMBER":    {"UNIQUE": 9},
          "RESOLVED-IP-ADDRESS": {"UNIQUE": 1}}
36629502 {"EMAIL":               {"UNIQUE": 180, "DUPLICATE": 171},
          "DOMAIN":              {"UNIQUE": 1, "DUPLICATE": 437},
          "CREDENTIAL":          {"UNIQUE": 145111, "DUPLICATE": 169751},
          "BANK-CARD-NUMBER":    {"UNIQUE": 7, "DUPLICATE": 7},
          "RESOLVED-IP-ADDRESS": {"UNIQUE": 5}}
```

Configuration overview

- Limited DSL (domain-specific language)
- Can configure destinations (”sinks”)
  - Web dashboard
  - Email
  - MISP
- Filters specify criteria and destination for matching artefacts

Configuration overview

- Limited DSL (domain-specific language)
- Can configure destinations (”sinks”)
  - Web dashboard
  - Email
  - MISP
- Filters specify criteria and destination for matching artefacts
- Model quite simplistic
  - Will change in the future
  - Need more use-cases

Email sink example

```
(define-sink email (smtp-sink)
  (:server "smtp.your.org")
  (:from "pastelyzer@your.org")
  (:subject (extract artefact-summary-by-class))
  (:body "URL: " (extract local-url) :fl
         "Origin: " (extract remote-url) :fl
         "Origin (raw): " (extract remote-raw-url) :fl
         (extract artefact-descriptions))
  (:recipients "pastelyzer-notifications@your.org"))
```

MISP sink example

```
(define-sink local-misp (misp-sink)
  (:server "https://127.0.0.1:5000/")
  (:api-key (env "MISP_API_KEY"))
  (:ca-cert (or (env "MISP_CA_CERT")
                "misp/ca.pem"))
  (:user-cert (or (env "MISP_USER_CERT")
                  "misp/misp.crt.pem"))
  (:user-key (or (env "MISP_USER_KEY")
                 "misp/misp.key.pem"))
  (:user-key-pass (env "MISP_USER_KEY_PASS")))
```

MISP sink example (cont.)

```
(define-sink misp-cc-event (local-misp)
  (:alert yes)
  (:publish yes)
  (:title "Credit card(s) detected")
  (:sharing-group "Guys with money")
  (:document-action (add-tags "CardFraud" "tlp:amber"))
  (:document-action (add-attribute :type "url"
                                   :category "External analysis"
                                   :value (extract source-url)))
  (:item-action (add-attribute :type "cc-number"
                               :category "Financial fraud"
                               :value (extract digits)
                               :comment (extract note))))
```

Simple filter

```
(define-artefact-filter important-cc
    (type? important-card-number)
  (collect-into misp-cc-event)
  (collect-into email))
```

More involved filter

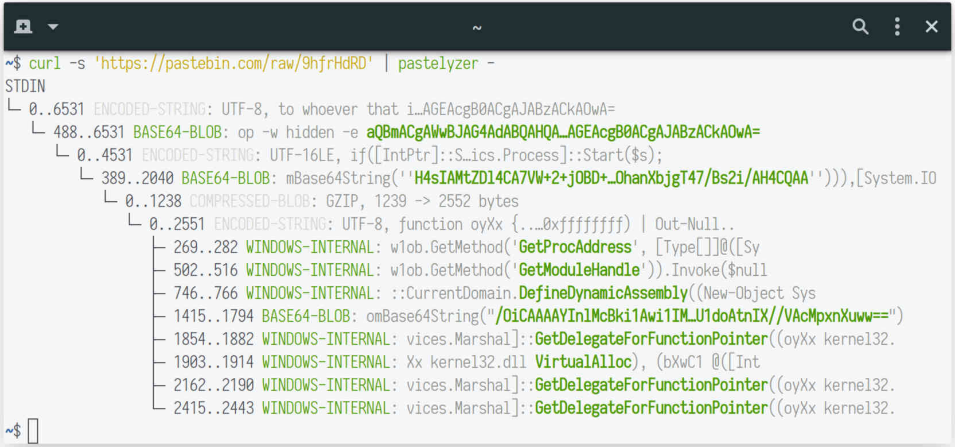
```
(define-artefact-filter demo
    ;; Everything except playlist entries and PNG images.
    (not (or (type? m3u-entry)
             (and (type? embedded-binary)
                  (-> (extract embedded-binary-bytes)
                      (starts-with? [89 50 4E 47])))))
  (collect-into dashboard))
```

- The prefix bytes are specified only once
- Does not matter how the bytes were encoded

Outline

CERT.LV

Screenshot or didn't happen! Interactive demo?



```
~$ curl -s 'https://pastebin.com/raw/9hfrHdRD' | pastelyzer -
STDIN
└ 0..6531 ENCODED-STRING: UTF-8, to whoever that i…AGEAcgB0ACgAJABzACkAOwA=
  └ 488..6531 BASE64-BLOB: op -w hidden -e aQBmACgAWwBJAG4AdABQAHQA…AGEAcgB0ACgAJABzACkAOwA=
    └ 0..4531 ENCODED-STRING: UTF-16LE, if([IntPtr]::S…ics.Process]::Start($s);
      └ 389..2040 BASE64-BLOB: mBase64String(''H4sIAMtZDl4CA7VW+2+jOBD+…OhanXbjgT47/Bs2i/AH4CQAA''))),[System.IO
        └ 0..1238 COMPRESSED-BLOB: GZIP, 1239 -> 2552 bytes
          └ 0..2551 ENCODED-STRING: UTF-8, function oyXx {.…0xffffffff) | Out-Null..
            ├ 269..282 WINDOWS-INTERNAL: w1ob.GetMethod('GetProcAddress', [Type[]]@([Sy
            ├ 502..516 WINDOWS-INTERNAL: w1ob.GetMethod('GetModuleHandle')).Invoke($null
            ├ 746..766 WINDOWS-INTERNAL: ::CurrentDomain.DefineDynamicAssembly((New-Object Sys
            ├ 1415..1794 BASE64-BLOB: omBase64String("/OiCAAAAYInlMcBki1Awi1IM…U1doAtnIX//VAcMpxnXuww==")
            ├ 1854..1882 WINDOWS-INTERNAL: vices.Marshal]::GetDelegateForFunctionPointer((oyXx kernel32.
            ├ 1903..1914 WINDOWS-INTERNAL: Xx kernel32.dll VirtualAlloc), (bXwC1 @([Int
            ├ 2162..2190 WINDOWS-INTERNAL: vices.Marshal]::GetDelegateForFunctionPointer((oyXx kernel32.
            └ 2415..2443 WINDOWS-INTERNAL: vices.Marshal]::GetDelegateForFunctionPointer((oyXx kernel32.
~$ 
```

Current state

- Useful for playing around
    - Sadly not much else at the moment
- Enhancements planned
    - JSON output
    - Artefact extraction to files (à la `binwalk`)
    - Feedback on use-cases welcome!

CERT.LV

Outline

Shortcomings

- As mentioned: work in progress
- Regular expressions still being used
- Quite a few false positives
    - This is intentional so we can deal with them

Shortcomings

- As mentioned: work in progress
- Regular expressions still being used
- Quite a few false positives
    - This is intentional so we can deal with them
- No logo

Upcoming features

- Integration with additional tools (at least Cuckoo)
- Ability to pass artefacts to external programs (e.g., yara)
- Storing more information in the database
  - Some artefact classes (e.g., IP addresses, domains, emails)
  - Full-text index
  - Extracted embedded artefacts
- Additional artefact classes
  - Crypto-currency wallets
  - IBAN account numbers
  - Phone numbers
- Make command-line interface actually useful
- REST API

What now?

- https://github.com/cert-lv/pastelyzer/
- Binary release for Linux
    - Additional platforms possible (for technically inclined)

Thank you!

janis.dzerins@cert.lv
https://www.cert.lv
certlv certlv