Pastelyzer
The paste analyzer

Jānis Džeriņš
1 Introduction

2 Technical details

3 Command line interface

4 Final remarks
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
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
1 Introduction

2 Technical details

3 Command line interface

4 Final remarks
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
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")))
(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

1. Introduction
2. Technical details
3. Command line interface
4. Final remarks
Screenshot or didn't happen! Interactive demo?
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!
Outline

1 Introduction
2 Technical details
3 Command line interface
4 Final remarks
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/](https://github.com/cert-lv/pastelyzer/)
- Binary release for Linux
  - Additional platforms possible (for technically inclined)