Probability Model Transforming Encoders Against Encoding Attacks

Haibo Cheng[1], Zhixiong Zheng[1], Wenting Li[1],
Ping Wang[1], Chao-Hsien Chu[2]

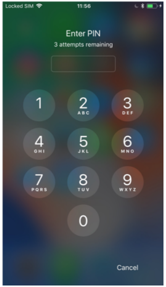
[1]Peking University, [2]Pennsylvania State University

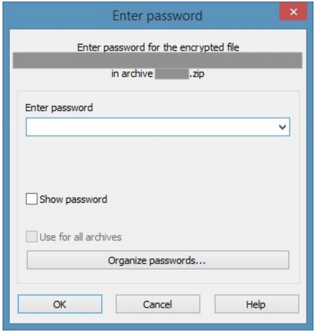
August 16, 2019 @ USENIX Security

Password-based encryption (PBE)

Fundamental scheme for:

1. Authentication
   a. PC, mobile phone, or Internet service.
2. Encryption
   a. Disk encryption
      i. FileVault on macOS
      ii. BitLocker on Windows
   b. File encryption
      i. VeraCrypt/TrueCrypt
      ii. Zip





Password-based encryption (PBE)

The key is password, different from cryptographic key

1. Human-generated and memorable.
2. Easy to be cracked.

Traditional Countermeasures:

1. Increase the complexity of decryption
   a. Salt.
   b. Use special password-hashing functions: iterated hash functions, memory-hard functions. Disadvantage: *increasing legitimate users' cost by the same factor.*
2. Harden passwords with other factors
   a. Biometric factor: fingerprint, iris, keystroke.
   b. Device: smart card, smart phone, server.

   Disadvantage: *worse on deployability; the encrypted message cannot be recovered, if the factors get stolen or lost.*

Honey encryption (HE)

A novel countermeasure proposed on EUROCRYPT'14

1. Idea: generate decoy messages for incorrect passwords/keys to confuse attackers.
2. Advantage: not increase the users' cost; not decline on deployability; significantly improve security.
3. Method: distribution transforming encoder (DTE)
   - a. Encrypt: Encode then encrypt
     - i. First encode the message M to a seed S by DTE.
     - ii. Then encrypt S by traditional PBE.
   - b. Decrypt: Decrypt then decode
     - i. With the right key K, yield the right S and M.
     - ii. With a wrong key K', yield a randomly wrong S' and M'.

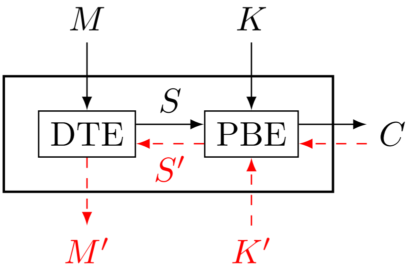
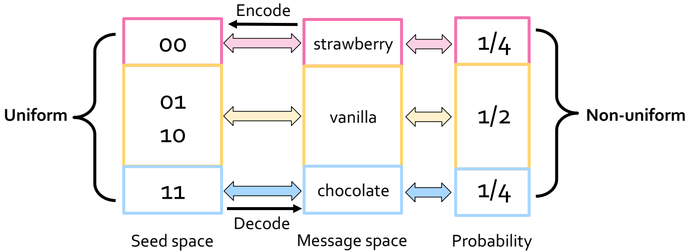


Figure 1: Honey encryption

Distribution transforming encoder (DTE)

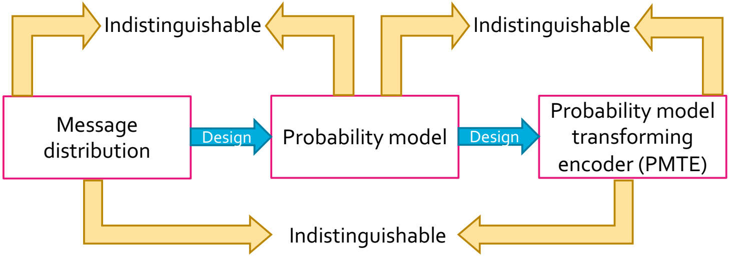
IS-DTE

1. Proposed on EUROCRYPT'14 [1]
2. For messages following simple distributions, e.g., uniform distributions, normal distributions.
3. Method: inverse sampling.



Probability model transforming encoder (PMTE)

Great challenge to design DTEs for messages following intricate distributions

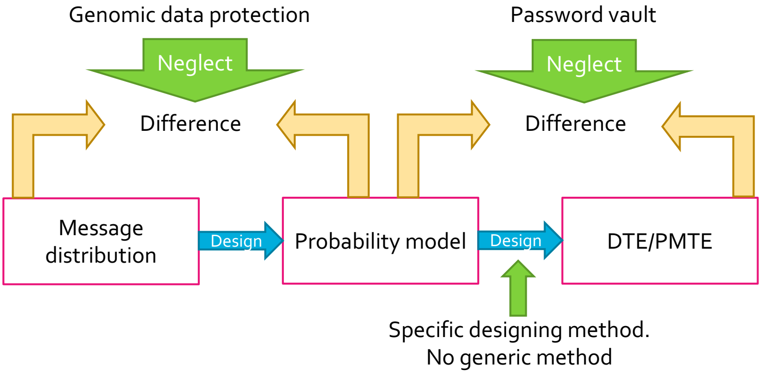


Existing PMTEs

1. Two for password vaults: NoCrack (S&P'15) [2] and Golla et al.'s scheme (CCS'16) [3].
2. One for genomic data protection: GenoGuard (Huang et al., S&P'15) [4].

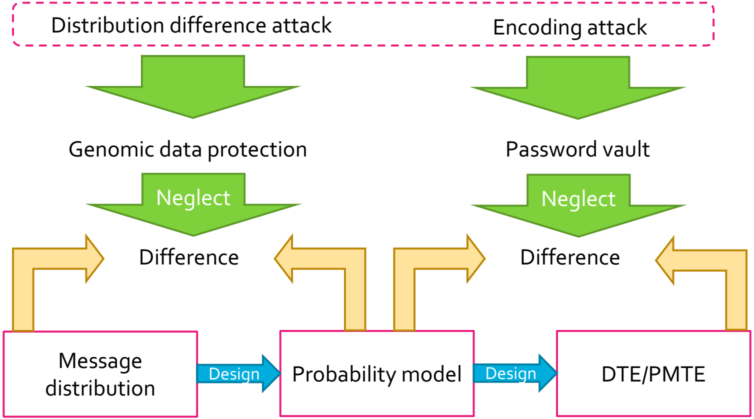
Gap in existing research

The security analysis is not comprehensive



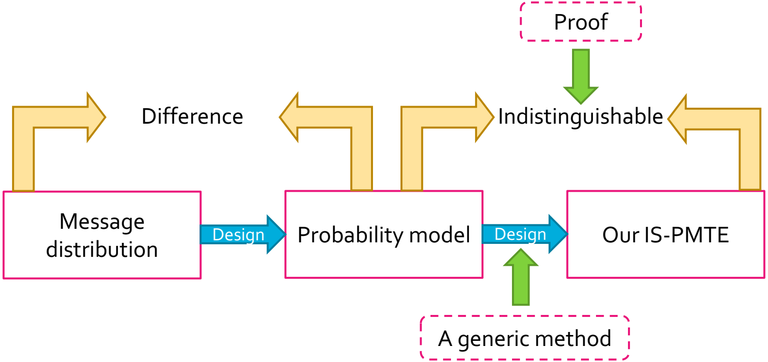
Our work

Two kinds of attacks



Our work

A generic designing method for PMTEs



Attacker model

Attacker's ability

1. Steal the storage file, i.e., ciphertext.
2. Know the PBE (encrypt/decrypt algorithm) and DTE/PMTE
3. Enumerate all keys offline.
4. Know some statistics about real messages (not needed for encoding attacks).
5. (For password vault) can perform a certain number of online verifications.

Attacker's goal

Distinguish the real message from a large number of decoy messages.

Attacker model

Attacker's process

1. Enumerate all keys and yield a large number of messages (only one of them is real).
2. To distinguish the real one
   1. For password vaults, sort the messages by some means and verify them online.
   2. For genomic data, just guess one offline.

   Formalization: Sort the message in decreasing order of a weight function p.
   The weight $p(M)$ usually reflects the probability that M is real.

Security

1. Only focus on the security of PMTEs: the distinguishability between the real and decoy messages.
2. Do not consider the security of keys: the strength of passwords.

Attacker model

Security metrics

1. The rank of the real messages in relative form i.e., real numbers in $[0, 1]$.
   (E.g., in 1000 decoy messages, 30 rank in front of the real one, then the rank is 0.3 .)
2. The rank cumulative distribution function $F(x)$.
3. The average rank $\overline{r}$.

$$\overline{r} = 1 - \int_0^1 F(x)\,dx$$

4. Accuracy α, the probability that the attacker distinguishes the real one between one real message and one decoy message.

$$\alpha = 1 - \overline{r} = \int_0^1 F(x)\,dx$$

Attack against GenoGuard

Genomic data

1. Single nucleotide variant (SNV) sequence represented by a string with $\{0, 1, 2\}$ alphabet.
2. Real dataset: 165 individuals' SNV sequences of length 1000.
3. Decoy data: generated by decoding random seeds with the PMTE.

Our attack: A classifier PCA+SVM

Principal component analysis (PCA) for dimensionality reduction (from 1000 to 10).
Support vector machine (SVM) for classification.

1. Training:
   1. Randomly pick half of real SNV sequences and generate the same number of decoy SNV sequences for training.
   2. Train PCA model and SVM in turn.

Attack against GenoGuard

Test/Attack

1. Use the rest real sequences for test.
2. Calculate the ranks of real sequences and other metrics (generate 999 decoy sequence for each real one).
3. The weight $p_{\mathrm{PCA+SVM}}$ for a sequence is the SVM-estimated probability that the dimension-reduced sequence is real.

Experimental results

Even for recombination model, 76.54% accuracy and 47.88% ($F(0)$) individuals' real sequences rank first.

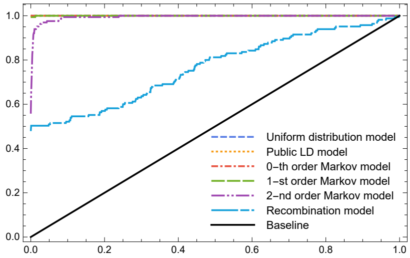


Figure 2: The rank cumulative distribution function

PMTE/Probability model	$\overline{r}$	α	$F(0)$	$F^{-1}(1)$
Uniform distribution model	0.00%	100.00%	100.00%	0.00%
Public LD model	0.00%	100.00%	99.39%	0.20%
0-th order Markov model	0.00%	100.00%	100.00%	0.00%
1-st order Markov model	0.01%	99.99%	99.39%	1.30%
2-nd order Markov model	0.53%	99.47%	55.76%	23.92%
Recombination model	23.46%	**76.54%**	**47.88%**	99.90%

Attacks against password vault schemes

Password vault

1. Store one user's multiple passwords on different websites/services.
2. These passwords are usually weak and similar.

NoCrack

1. PCFG model: characterize the single password distribution
   A password "password1": $\texttt{S} \to \texttt{WD}, \texttt{W} \to \text{password}, \ \texttt{D} \to 1$
2. Sub-grammar: characterize the password similarity
   A vault $V = (\text{password}, \text{password1})$,
   its sub-grammar $SG = \{\texttt{S} \to \texttt{W}, \ \texttt{S} \to \texttt{WD}, \ \texttt{W} \to \text{password}, \ \texttt{D} \to 1\}$
3. Encode
   1. Parse V's sub-grammar SG, encode SG;
   2. Encode each password in V based on SG;
   3. Concatenate all seeds and output the concatenation.

Attacks against NoCrack

Defects in NoCrack

A sub-grammar for a real vault is parsed from the vault, but a sub-grammar for a decoy vault is generated randomly. This leads:

1. There definitely exists no unused rule in sub-grammars for real vaults, but may exist for decoy vaults. Feature UR.
2. There definitely exists no duplicate rules in sub-grammar for real vaults, but may exist for decoy vaults. Feature DR.

Attack	$\overline{r}$	α	$F(0)$	$F^{-1}(1)$
Feature UR attack	15.14%	84.86%	0.36%	42.24%
Feature DR attack	26.96%	73.04%	0.00%	54.95%

Golla et al.'s scheme

Similar defects.

Encoding attacks

Encoding attacks

These feature attacks *do no need any statistics about the real distribution and only exploit the DTE/PMTE*. We name such attacks *encoding attacks*.

Questions:

1. Why these PMTEs cannot resist encoding attacks?
2. Is there other features?
3. What is the principle for encoding attacks?

To answer the questions:

1. First formalize the probability models into a unified form.
2. Idea: A model usually designs a series of generating rules to assign messages probabilities. The probability of a message is the probability that it is generated by the rules.

Generative probability model

Definition

A *generative probability model (GPM)* is a 5-tuple $(\mathcal{M}, \mathcal{R}, \mathcal{RS}, G, P)$:

1. $\mathcal{M}$ is the message space,
2. $\mathcal{R}$ is the set of generating rules,
3. $\mathcal{RS} \subset \mathcal{R}^*$ is the set of valid sequences of generating rules,
4. G is the generating function mapping a sequence in $\mathcal{RS}$ to a message in $\mathcal{M}$,
5. P is the probability density function on $\mathcal{RS}$.

Here $\mathcal{M}, \mathcal{R}, \mathcal{RS}$ are finite sets, G is surjective. Then the probability density function P on $\mathcal{M}$ is given as

$$P(M) = \sum_{RS \in G^{-1}(M)} P(RS). \quad (1)$$

If for every message, there only exists one generating sequence (i.e., G is bijective), then the GPM is *unambiguous*, and otherwise, it is *ambiguous*.

Formalization

PCFG model

1. A generating rule is a production rule in PCFG.
2. A generating sequence is a (leftmost) derivation of a string.
3. $P(r_i|r_1 r_2 \ldots r_{i-1}) = P(r_i)$.

Sub-grammar

A generating sequence of the sub-grammar $\{\texttt{S} \rightarrow \texttt{D}, \texttt{S} \rightarrow \texttt{W}, \texttt{D} \rightarrow 123456, \texttt{W} \rightarrow \text{password}\}$ is $(\#\texttt{S} = 2, \texttt{S} \rightarrow \texttt{D}, \texttt{S} \rightarrow \texttt{W}, \#\texttt{D} = 1, \texttt{D} \rightarrow 123456, \#\texttt{W} = 1, \texttt{W} \rightarrow \text{password})$.

Other models

Similar formalizations, e.g., Markov models, a generating rule is a character.

Generating graph

Generating graph: represent a GPM visually

1. A directed acyclic graph with a single source.
2. An edge represents a generating rule.
3. A sink represents a message.
4. A path from the source to a sink represents a generating sequence, called generating path.

The principle of encoding attacks/Defects in existing password vault schemes

1. The ambiguous probability models.
2. But only choose a deterministic path when encoding.

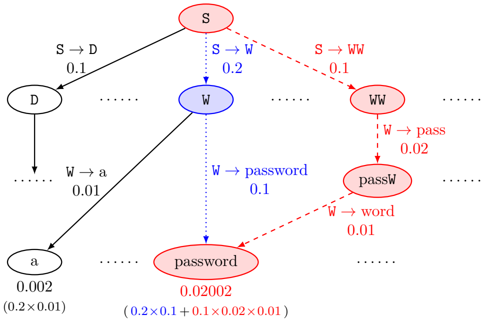


Figure 3: Generating graph for a PCFG model

Generic encoding attacks

Generic encoding attacks

1. Weak encoding attack: exclude these seeds whose paths cannot be chosen when encoding.
2. Strong encoding attack: sort the rest seeds by $\frac{1}{P(RS)}$.

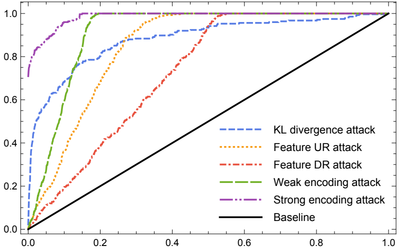


Figure 4: Chatterjee et al.'s PMTE [2]

Attack	$\overline{r}$	α	$F(0)$	$F^{-1}(1)$
KL divergence attack	11.83%	88.17%	1.82%	98.80%
Feature UR attack	15.14%	84.86%	0.36%	42.24%
Feature DR attack	26.96%	73.04%	0.00%	54.95%
Weak encoding attack	8.74%	91.26%	0.36%	19.42%
Strong encoding attack	1.44%	**98.56%**	70.55%	15.02%

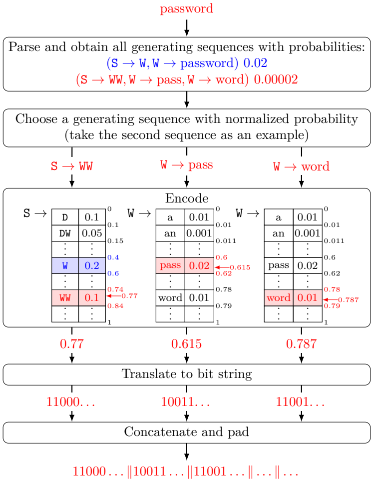
A generic method for designing PMTEs

Conditional DTEs

IS-CDTE: For each condition X, construct **IS-DTE**$_X$ according to the conditional distribution $P(M|X)$.

Our **IS-PMTE**

1. Encode M:
   a. Parse all generating sequence $G^{-1}(M)$, and choose one RS with its probability.
   b. Encode each rule r_i in RS to S_i by using **IS-CDTE** on condition $(r_1, r_2, \ldots, r_{i-1})$.
   c. Concatenate S_i, pad the concatenation to a fixed length, and output the result S.



The security of our IS-PMTEs

We prove

Our IS-PMTE is indistinguishable from the corresponding GPM.

Experimental results under the strong encoding attack

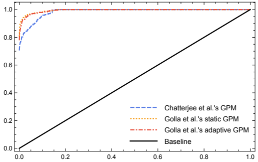


Figure 5: Original PMTEs

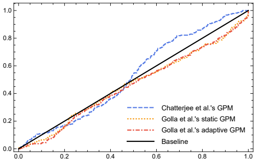
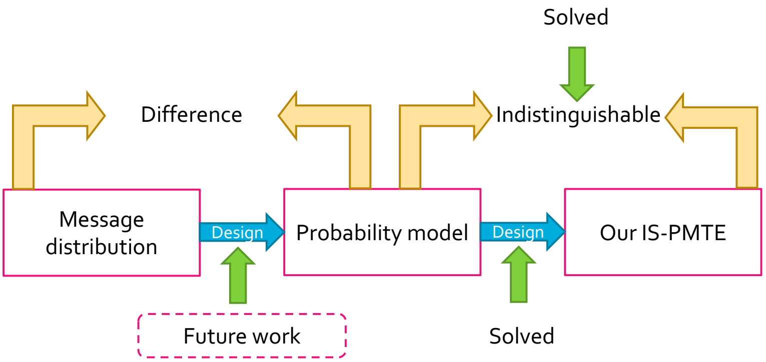


Figure 6: Our IS-PMTEs

Probability model	Accuracy α Original	Accuracy α Our
Chatterjee et al.'s GPM	98.56%	52.56%
Golla et al.'s static GPM	99.52%	46.38%
Golla et al.'s adaptive GPM	99.42%	45.75%

Future work

Design probability models



Q&A

Thank you