Alma Mater Studiorum · Università di Bologna

Scuola di Ingegneria e Architettura
Dipartimento di Informatica · Scienza e Ingegneria · DISI
Corso di Laurea in Ingegneria Informatica

TITOLO DELLA TESI

Relatore:
Prof.

Correlatore:
Prof.

Presentata da:
Karina Chichifoi

Anno Accademico 2019/2020

Dedica dedicosa.
— A capo.

Indice

Elenco delle figure

Introduzione

1 | Stato dell'arte

La sicurezza delle password al giorno d'oggi riveste un ruolo significativo nel garantire confidenzialità e integrità dei dati personali degli utenti e delle aziende. Solitamente si è più propensi a scegliere password semplici da ricordare, come riferimenti autobiografici, oppure sequenze di caratteri molto comuni (e.g. `qwerty`, `123456`). Per semplificare la memorizzazione, si utilizzano spesso passsword brevi, in media da 9-10 caratteri e composte in gran parte da caratteri minuscoli. [1]

Tuttavia questa scelta porta a maggiori probabilità di subire violazioni dei propri account, poiché password semplici sono vulnerabili ad attacchi di forza bruta. Inoltre, tramite tecniche di ingegneria sociale, è possibile individuare il criterio di scelta dell'utente, eventualmente ragionando sui dati disponibili grazie ai data breach.

Nel 2020 sono stati confermati 3950 data breach, dal costo medio di 3,86 milioni di dollari. Il 52% dei breach è stato causato da attacchi informatici e il numero di giorni medio per individuare un breach è stato di 207 giorni. [2] Il 42% è causato da attacchi su applicazioni web e il il metodo più comune di attacco (82%) ha utilizzato credenziali rubate o ottenute tramite attacchi a forza bruta. Il 58% dei breach conteneva dati personali. [3]

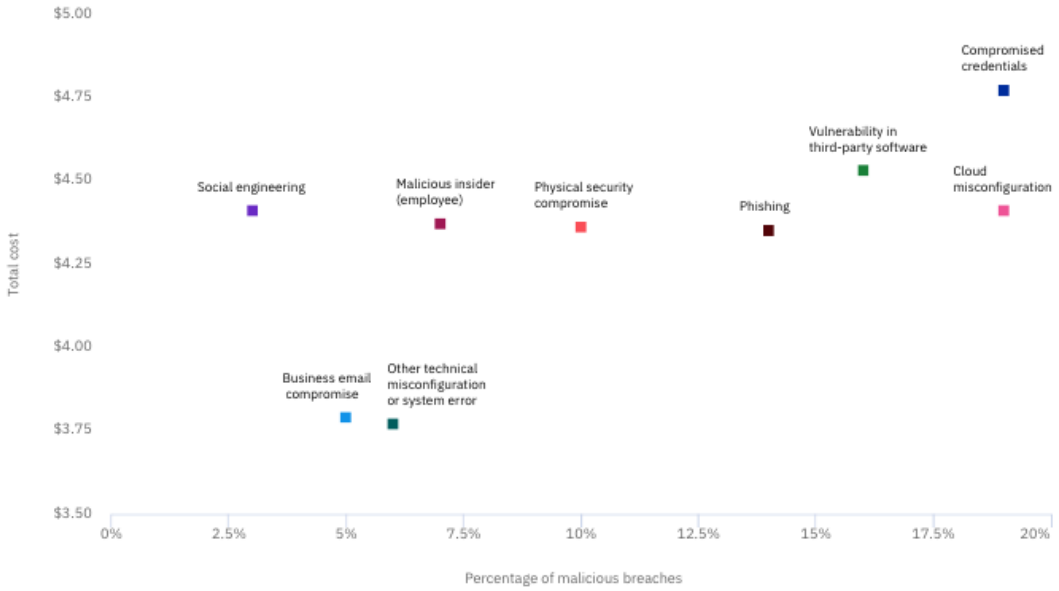

Figura 1.1: Costo medio e frequenza di data breach causati da attacchi informatici, in base alla causa, nel 2020

Sebbene la maggior parte delle password siano crittografate, è possibile risalire alla forma testuale mediante strumenti come ***Hashcat*** e ***John The Ripper***.

In seguito alla diffusione delle credenziali, gli utenti decidono di cambiare password e la scelta ricade spesso su varianti usate su altri account.

1.1 Scelta di una nuova password

L'utente medio ha la tendenza a scegliere password semplici. Per questo motivo, spesso la nuova password è il risultato di una leggera variazione della vecchia password, o una combinazione di password precedenti. [4]

Un modo per verificare la sicurezza della password è utilizzare strumenti come ***zxcvbn***, che riesce a riconoscere:

- 30000 password comuni;
- nomi e cognomi comuni negli USA;
- parole spesso utilizzate in inglese su Wikipedia;
- parole spesso utilizzate alla televisione e film statunitensi;
- date;
- ripetizioni di lettere (`aaaa`);
- sequenze alfabetiche (`abcde`);
- sequenze di tastiera (`qwertyuiop`);
- il codice `l33t`.

Altri strumenti, come ***Kapersky password checker***, controllano anche dati di numerosi data breach raccolti da *Have I been Pwned?*. Questi approcci, tuttavia, non controllano la cronologia delle password di uno specifico utente, ma soltanto la resistenza ad attacchi di forza bruta.

Per questo motivo sono state studiate strategie che tengono conto delle credenziali utilizzate. Alcune sfruttano un approccio probabilistico, come i ***Bloom Filter***, che...*TODO* . Un'altra possibile modalità utilizza ***Word Embedding*** di password.

1.2 Bloom filters

TODO

1.3 Word embedding

Per capire il contesto delle parole e per poterle rappresentare in base alla sfera semantica e alla sintassi, si ricorre un insieme di tecniche che prevedono il *mapping* delle parole o delle frasi di un dizionario in vettori di numeri reali, note come ***Word Embedding***. Parole simili possiedono una codifica simile.

Per stabilire il valore di ogni *embedding* si allena una rete neurale con specifici parametri e le dimensioni variano tra 8 (per piccoli dataset) a 1024. Maggiore è la dimensione di un embedding, maggiore risulta la quantità di informazioni relativa alle relazioni tra parole. [5]

1.3.1 Similarità tra parole

Per potere stabilire se due parole appartengono alla stessa sfera semantica si utilizza un metodo noto come ***cosine similarity***.

$$similarity = cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|} \tag{1.1}$$

Due parole risultano simili quando il valore del coseno è 1, ovvero quando l'angolo tra i due vettori risulta nullo. Si consideri il seguente esempio:

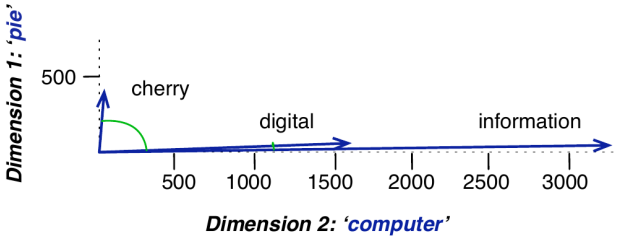

Figura 1.2: Esempio di calcolo della similarità tra word embedding

Nella figura sono mostrati i vettori di 3 parole (*cherry*, *digital* e *information*) in uno spazio bidimensionale definiti dal numero di occorrenze in vicinanza alle parole *computer* e *pie*. Notare che l'angolo tra *digital* e *information* risulta minore rispetto all'angolo tra *cherry* e *information*. Quando due vettori risultano più simili tra loro, il valore del coseno risulta maggiore, ma l'angolo risulta minore. Il coseno assume valore massimo 1 quando l'angolo tra i due vettori risulta nullo (0°); il coseno degli altri angoli risulta inferiore a 1. [6]

1.3.2 Tipologie di word embedding

Word2Vec

Word2Vec è un insieme di modelli architetturali e di ottimizzazione utilizzati per imparare ***word embedding*** da un vasto corpus di dati, sfruttando reti neurali.
Un modello allenato con Word2Vec riesce a individuare le parole simili tra loro, in base al contesto, grazie alla *cosine similarity* esaminata precededentemente.

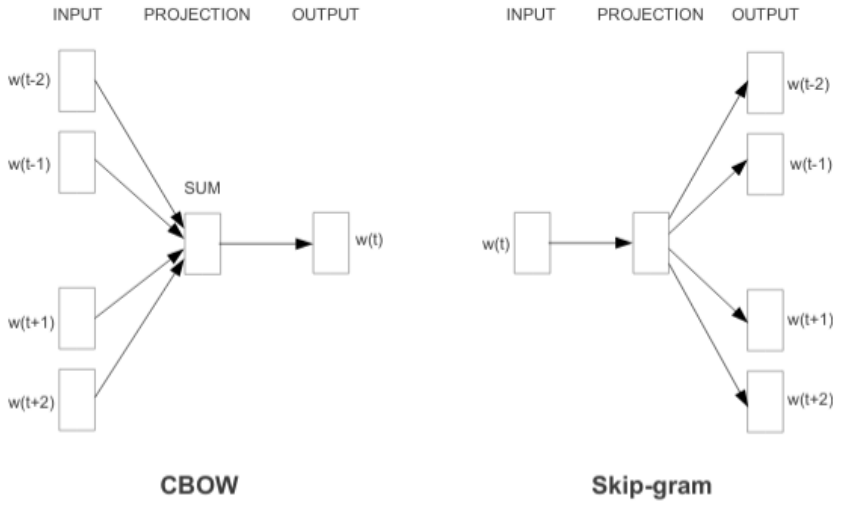

Figura 1.3: CBOW vs skip-gram:

Word2Vec utilizza due modelli di architetture:

- ***CBOW*** (continuous bag of words): l'obiettivo del training è combinare le rappresentazioni delle parole limitrofe per prevedere la parola centrale.
- ***Skip-gram***: simile a CBOW, con la differenza che viene utilizzata la parola centrale per prevedere le parole circostanti relative allo stesso contesto.

CBOW risulta più veloce ed efficace in caso di dataset di grandi dimensioni, tuttavia, nonostante la maggiore complessità, ***Skip-gram*** è in grado di trovare parole mai viste, per dataset di minori dimensioni. [7]

FastText

FastText è una libreria open-source proposta da Facebook che estende Word2Vec, e consente un apprendimento efficiente di rappresentazioni di parole e di classificazioni di frasi. Anziché allenare un modello fornendo ogni singola parola di un dataset, FastText prevede l'apprendimento tramite *n-gram* di ciascuna parola. Un esempio per capire cosa siano effettivamente gli n-gram è il seguente:

$$ciao = \{\{c, i, a, o\}, \{ci, ia, ao\}, \{cia, iao\}, \{ciao\}\}$$

In questo esempio, con $n\_mingram = 1$ e $n\_maxgram = 4$, *ciao* viene espresso come l'insieme di tutte le sottostringhe di lunghezza minima pari a 1, e lunghezza massima pari a 4.

FastText consente di ottenere, con più probabilità rispetto a Word2Vec, parole ***out-of-dictionary***, ovvero parole sconosciute al modello in fase di training.

2 | Analisi progettuale

3 | Implementazione

4 | Risultati

Conclusioni

Conclusione.

Ringraziamenti

Ringraziamenti.

Bibliografia

[1] Sarah Pearman et al. «Let's Go in for a Closer Look: Observing Passwords in Their Natural Habitat». In: ***Commun. ACM*** 50.1 (ott. 2017), pp. 295–310. ISSN: 1557-735X. DOI: 10.1145/3133956.3133973. URL: https://dl.acm.org/doi/pdf/10.1145/3133956.3133973.

[2] ***Cost of a Data Breach Report 2020***. 2020. URL: https://www.ibm.com/security/digital-assets/cost-data-breach-report/#/pdf.

[3] ***Data Breach Investigations Report 2020***. 2020. URL: https://enterprise.verizon.com/resources/executivebriefs/2020-dbir-executive-brief.pdf.

[4] ***Password Usage Study: How Do We Use Passwords?*** 2019. URL: https://web.archive.org/web/20200610025025/https://www.hypr.com/wp-content/uploads/password_usage_study_infographic_hypr.png.

[5] ***Word embeddings***. 2021. URL: https://www.tensorflow.org/tutorials/text/word_embeddings.

[6] Daniel Jurafsky e James H. Martin. ***Speech and Language Processing***. 2020. URL: https://web.stanford.edu/~jurafsky/slp3/ed3book_dec302020.pdf.

[7] Tomas Mikolov et al. ***Efficient Estimation of Word Representations in Vector Space***. 2013. arXiv: 1301.3781 [cs.CL].