CERIAS

The Center for Education and Research in Information Assurance and Security

Duress Authentication via Partially Homomorphic Encryption

Ben Harsha, Mikhail Atallah

Overview

- **Duress Authentication:** An authentication system that allows users to send duress signals
- Might be used in places like banks as a sort of silent alarm
- Current systems have fatal flaws or require a third party to check all logins
- Current systems are either hard to use or easy to bypass
- We develop a new method to allow duress authentication
- Simple implementation by changing password storage method

Duress Adversaries



Partially Homomorphic Encryption



- Homomorphic encryption allows for operations to be performed on data *while it is encrypted*
- Fully Homomorphic Encryption: Any operation can be performed on encrypted data
 - Currently very inefficient!
- **Partially Homomorphic Encryption:** Only some operations can be performed
 - Limited, but efficient
 - E.g. $Dec_{sk}(Enc_{pk}(x) * Enc_{pk}(y)) = x*y$

...

We consider adversaries who are physically forcing someone to reveal their password, whether via force, blackmail, or some other threat. In an attack this adversary:

- Knows the system
- **Compromises multiple administrators**
- Monitors all network traffic
- Has full system access
- Can request as many responses from those they are attacking as they like
- Cautious They will never perform an action that is sure to trigger a duress signal

The Failings of Current Duress Systems

- Many claims to offer duress authentication
- Included in many US Patents, often as part of a larger system
- All include some flaw that makes them impractical or vulnerable
- They fall into two categories:
- Two Passwords: Each user has two passwords. One normal and one duress.
 - People struggle to remember passwords
 - Adversary can guess an win 50% of the time
- **Modified Passwords:** A user modifies their password in some predictable way
 - Type in a random number at the end vulnerable to typos
 - Move first letter to the end very obvious

Securepassword123 ecurepassword123S \rightarrow

A standard password transforms into an obvious duress signal

HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS \$5 WRENCH UNTIL HE TELLS US THE PASSWORD.



A New Duress System

Given some partially homomorphic public-key cryptosystem with a homomorphism over a group G with operation • we are able to store extra information in a password file to handle duress signals

- We store $Enc_{pk}(r_i)$, $Enc_{pk}(kw_i^{-1})$ in addition to the usual password data4
- Password storage now looks as shown below, with new additions in the green columns

| Username | Salt | H(salt + pwd) | Enc _{pk} (r _i) | $Enc_{pk}(kw_i^{-1})$ |
|----------|------------|---------------|-------------------------------------|-----------------------|
| alice | dc433a1f34 | 8f8ff2239b | 442d50f7e4 | 7f1eb7e075 |
| bob | a734ac7cb6 | 72ff38eb2d | 9c670ba400 | 8074ed62c9 |

...

The Login Process

. . .

1. The user enters their password

...

- 1. Check if H(pwd,salt) matches the stored data. If so continue, else reject
- 2. The user enters a keyword, kw'
 - 1. If kw' is not in the dictionary reject (probably a typo!)
 - 2. If it is, overwrite their password file record as follows

1. $\operatorname{Enc}_{pk}(r_i) \leftarrow \operatorname{Enc}_{pk}(r_i) \bullet (\operatorname{Enc}_{pk}(kw_i^{-1}) \bullet \operatorname{Enc}_{pk}(kw'))$

Our Requirements

For a useful duress password system we require the following features or properties



- Easy to use: The system should require minimal learning on the user's end, and should not require a large amount of memorization
- **Undetectable:** The system should not leak a duress signal to someone who does not have authorization to read it
- Low false-positive rate: We should ensure that false alarms are not a common occurrence
- Low false-negative rate: Adversaries should not be able to bypass the system
- **Spyware-resistant**: A system compromised by spyware before any signal is sent should not be able to reveal who sends a signal

The Keyword System

- Previously proposed by Atallah and Stefanov in 2010² this provides a method for sending a duress signal that is easy for users.
- Don't memorize two passwords.
- Memorize a keyword from a dictionary
- Your assigned keyword is for non-duress logins
- Any other keyword is a duress signal
- Keyword dictionary should have high edit distance to avoid typos (e.g. birds)
- This system satisfies the easy-to-use and low false positive/negative rates

- The overwriting process is key. If kw' == kw_i then the encrypted value does not change
- If kw' does not match kw_i, then $Enc_{pk}(r_i)$ is overwritten with an incorrect value
- Randomization within the cryptosystem prevents attacker from telling if anything changed

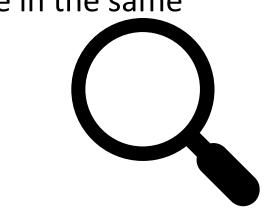
Checking for Duress

- Some Duress Authority holds the private keys to the system
 - Ideally external e.g. police, alarm company, but can be in the same organization as well if desired
- Authority stores the correct **r**_i value for each user
- To check for duress, compare the **r**_i value with the stored value
 - A mismatch is a duress signal!
- Optionally a global flag can be kept for quick polling as well
- At this point the duress response is up to the authority

Acknowledgements

- Thanks to Jeremiah Blocki for several conversations and feedback on this topic
- Portions of this work were supported by National Science Foundation Grant CPS-1329979 and by sponsors of the Center for Education and Research in Information Assurance and Security.
- Portions of this work were supported by a Rolls Royce Doctoral Fellowship grant
- The statements made herein are solely the responsibility of the authors.
- Modified from Randall Monroe's XKCD: Security https://xkcd.com/538/
- and Mikhail Atallah. "Duress detection for authentication attacks against multiple administrators." Proceedings of the 2010 ACM workshop on Insider threats. ACM, 2010.





...

