Title: Decrypting Phantom Wallets and Recovering Seed Phrase

Date: April 20, 2024, rev 2

Author: Cyclone

GitHub: https://github.com/cyclone-github

Credits: Blandyuk, for his assistance with research on nacl/secretbox encryption

Preface: These are the steps I have taken to successfully decrypt and recover the seed phrase from a Phantom crypto wallet. This method has been tested and verified on my own test wallet with a known password and seed phrase. For transparency, I am including the encrypted JSON strings, password, seed phrase, source code I wrote, and the steps taken, as this process can be replicated by following the steps outlined below.

Proof: The steps listed below provide proof that:
1. Phantom wallets can be decrypted
2. The seed phrase can be successfully recovered from the latest version of Phantom wallets as of this writing.

Phantom Wallet Version: **v24.6.0**
Wallet Password: "**password**"
Seed phrase: "**car odor pioneer ball treat build load famous develop surface detail art**"

1. Extract encrypted JSON strings
To extract the encrypted JSON string from the Phantom test wallet, a custom tool was written to extract the 4x encrypted JSON strings from the LevelDB where Phantom, as several other wallets such as Metamask and Atomic, store their vault data. On my test system running the Chrome browser on ParrotOS (debian based distro), the Phantom LevelDB was located at:

/home/$USER/.config/google-chrome/Default/Local\ Extension\ Settings/bfnaelmomeimhlpmgjnjophhpkkoljpa/

Example code used to extract first encrypted JSON, "**phantom-labs.encryption.encryptionKey**", which I'll refer to as "**encryptedKey**":
https://github.com/cyclone-github/phantom_pwn/tree/main/phantom_extractor

Similar code was written to extract the remaining 3x encrypted JSON strings.

Once the encryptedKey JSON was extracted, it somewhat resembled the previous version of Phantom vaults, and turns out, it uses the same encryption implementation.

Extracted encryptedKey:

{"encryptedKey": {"digest":"sha256","encrypted":"5pLvA3bCjNGYBbSjjFY3mdPknwFfp3cz9dCBv6izyyrqEhYCBkKwo3zZUzBP44KtY3","iterations":10000,"kdf":"pbkdf2","nonce":"NZT6kw5Cd5VeZu5yJGJcFcP24tnmg4xsR","salt":"A43vTZnm9c5CiQ6FLTdV9v"},"version":1}

2. Decrypting encryptedKey
To decrypt encryptedKey, a tool based off my previous vault decryptors was modified to work with the **nacl/secretbox** encryption implementation. This is the same encryption implementation Phantom used on the previous version of their wallet.

phantom_decryptor source code:
https://github.com/cyclone-github/phantom_pwn/tree/main/phantom_decryptor

After successfully decrypting encryptedKey, the payload was a raw string of bytes, and after testing, it was determined that the payload was used as the key for the remaining encrypted JSON strings. To make it possible to store the raw bytes key in my notes, the payload was encoded as base64 and listed below for reference:

fp+UEdf3RUtAqxtxEMjWQJKStHO9HnaoOoN7Robaj3k=

3. Decrypting the 3x remaining JSON strings
Using the decrypted encryptedKey payload as the key for the 3x remaining encrypted JSON strings, I was able to decrypt them and was greeted with 3x decoded JSON strings:

phantom-labs.vault.cache.seed:
{"version":1,"identifier":"7aec91735f7e125dde475e982aae316c88f336e361593f70a4e51c2e1c7c1dd7","seed":
{"bytes":"NBJYY0wt4wzvDY5qY+STVo78yIdPv5de2aa6yxKTcb69ZaqtKLb450J0ntcZkp1eMMfcrTgvlTJl/bRtyyyE0g==","encoding":"base64"}}

phantom-labs.vault.integrityCheck:
{"version":1,"isSafelyOnboarded":true,"lastCheckedTimestamp":1713562315056}
phantom-labs.vault.seed:
{"version":1,"identifier":"7aec91735f7e125dde475e982aae316c88f336e361593f70a4e51c2e1c7c1dd7","name":"Account 1","entropy":
{"0":34,"1":83,"2":38,"3":148,"4":136,"5":254,"6":124,"7":59,"8":160,"9":186,"10":149,"11":60,"12":155,"13":68,"14":241,"15":6}}

vault.cache.seed was a base64 string which, when decoded, was raw bytes.

vault.integrityCheck gave a version number, boolean of "true", and a timestamp.

vault.seed was a sequence of entropy, which, when used as the entropy for BIP39, produced the original 12 word seed phrase from the test wallet:

car odor pioneer ball treat build load famous develop surface detail art

Phantom's mysterious implementation of encryption and seed phrase obfuscation was finally pwnd.

~Cyclone

- **All JSON strings from vault:**
  - .phantom-labs.vault.cache.seed.7aec91735f7e125dde475e982aae316c88f336e361593f70a4e51c2e1c7c1dd7 | {"content":{"digest":"sha256","encrypted":"2T4Q9BZH3i3GbYMEoyhEWpt4H26Sb7TWukwT8ZHSbKJmCMvF1tx8DompQWrhW1YiWdt5z1whnRqfKgDgnzHJxGidRCZRZ3YFHKe8ARXjT21bnxb848UVCsTeERg5AFc3JZir8Kkh2eD2A1in8QUfN9XXLKJNPHpsoNwpxso6rgem47pBQcgxCSvMX4Q8EREu8mHXKLuKajTh8mvbwUrtms6Q4aNcKkRXxZGmvyW7EpEdwDhsuy18A5r5e6H5pQ6UJnkbWZHykcwQ89feGesJrkaU81yE51BYjHgYS1acpQe39mg9uvz43cnYT2DqPqHSbEWP","iterations":10000,"kdf":"pbkdf2","nonce":"5UtP4bwFzH4G9gtvpVyickxfWQMiTEoWc","salt":"A7bjaKUBtRLepfH8MFcV2u"},"version":1}
  - .phantom-labs.vault.seed.7aec91735f7e125dde475e982aae316c88f336e361593f70a4e51c2e1c7c1dd7| {"content":{"digest":"sha256","encrypted":"2z6wicLSo4mXEtkHnhHMZRqHAcRnZXSDnaebAg4tfNQWWP4NBGiywYgZNJGbSMpAJGq5sN1YFCZFuBxuiDLdPy4wckqmsHLipPSC7G8XiAQtH3TPi5THGjCSfciEj9jZ3BFfDCDjSX9x4X2kktxz2QMnQ8Usm5FTQmtw7QvDRt3xVmBRjKD5Q78UGE1d3NcUT64VZSuUHUS6Y5DzWidUKkbj7YGBu51MH6bNRsodqCVbPoSaistgP21f4giUWiMgFpDExAuq7oGkkAVFHB6FoqMebUvgpcaPCJy2BPEGjtU9jg3EwARqkReVWiJGXKCV77i5T84JGhW5qFyknz49L6qdPC9pVwsSMauiK7PRxgYe","iterations":10000,"kdf":"pbkdf2","nonce":"HbeN8rqQdM714DYro4McoxVJxmHnJnu4x","salt":"T83bLpyhG9ASg2atBmVwBN"},"version":1}
  - .phantom-labs.vault.integrityCheck | {"content":{"digest":"sha256","encrypted":"2BPiigXsTiqxBnj1mqoWvjw7eEbzfZspcUNAq892WNNzEkzNiFeezNth5etVLqJWNjZzqCeBduFGzDh4B12qZ4kdwwF1PPaDUoE7Uau4ooDzmbYSuRuZn8scknv12","iterations":10000,"kdf":"pbkdf2","nonce":"CSZ5xPChuG7x5W397C3ywdGv4c2VPAPaE","salt":"GNd7PJAMJZ3gUbc3arhvnt"},"version":1}

- **Source code for generating mnemonic seed phrase from entropy string:**

```
package main

import (
"fmt"

"github.com/tyler-smith/go-bip39"
)

/*
test code for decrypting Phantom vaults
by cyclone
https://github.com/cyclone-github/phantom_pwn
*/

func main() {
// entropy from phantom-labs.vault.seed
entropy := []byte{34, 83, 38, 148, 136, 254, 124, 59, 160, 186, 149, 60, 155, 68, 241, 6}

// convert byte slice to a mnemonic seed phrase using BIP39
mnemonic, err := bip39.NewMnemonic(entropy)
if err != nil {
fmt.Println("Error generating mnemonic:", err)
return
}

fmt.Println("Mnemonic (Seed Phrase):", mnemonic)
}
```

- **Source code for decrypting JSON strings:**

```
package main

import (
"crypto/sha256"
"encoding/base64"
"fmt"

"github.com/btcsuite/btcutil/base58"
"golang.org/x/crypto/nacl/secretbox"
"golang.org/x/crypto/pbkdf2"
)

/*
test code for decrypting Phantom vaults
by cyclone
https://github.com/cyclone-github/phantom_pwn
*/

func main() {
base64password := "fp+UEdf3RUtAqxtxEMjWQJKStHO9HnaoOoN7Robaj3k=" // decrypted,
base64-encoded password
passwordBytes, err := base64.StdEncoding.DecodeString(base64password)
if err != nil {
fmt.Printf("Failed to decode password from base64: %v\n", err)
return
}

saltBase58 := "" // salt from vault.cache.seed, vault.seed, vault.integrityCheck
nonceBase58 := "" // nonce from vault.cache.seed, vault.seed, vault.integrityCheck
encryptedBase58 := "" // encrypted from vault.cache.seed, vault.seed, vault.integrityCheck

salt := base58.Decode(saltBase58)
nonce := base58.Decode(nonceBase58)
encryptedData := base58.Decode(encryptedBase58)

if len(nonce) != 24 {
panic("Nonce must be exactly 24 bytes long")
}

key := pbkdf2.Key(passwordBytes, salt, 10000, 32, sha256.New)

var nonceArray [24]byte
copy(nonceArray[:], nonce)
var keyArray [32]byte
copy(keyArray[:], key)

decrypted, ok := secretbox.Open(nil, encryptedData, &nonceArray, &keyArray)
if !ok {
fmt.Println("Decryption failed")
return
}

if len(decrypted) < 16 {
fmt.Println("Decrypted data is too short")
return
}

fmt.Printf("%s\n", decrypted)
}
```