Information Disclosure

czego oko nie widzi, tego hackerowi nie żal

Co programista może zostawić w kodzie?

Zasadniczo... wszystko:

- sekrety
- informacje wyświetlające się w konsoli / na stronie
- nieobsłużone właściwie błędy
- domyślną konfigurację
- informacje o konfiguracji
- ...etc.

Jak programista zostawia takie dane?

- robots.txt
- backupy na serwerze
- informacje z bazy danych w wiadomości z błędem
- niepotrzebne ujawnianie nadmiernych danych
- hardkodowanie sekretów: kluczy API, haseł, etc...
- inna reakcja aplikacji w przypadku np. sprawdzania, czy login danego użytkownika istnieje

(https://portswigger.net/web-security/information-disclosure)

Co może wyciec?

- dane użytkowników
- dane biznesowe / firmowe
- techniczne dane o aplikacji / infrastrukturze, na której stoi

(https://portswigger.net/web-security/information-disclosure)

Co może wyciec?

- dane użytkowników (jakoś to obsługujemy, bo RODO)
- dane biznesowe / firmowe (jakoś to obsługujemy, bo BIZNES)
- techniczne dane o aplikacji / infrastrukturze, na której stoi (jakoś...)

Dlaczemu?

- trzeba dowozić nowe ficzery ("mitowanie dedlajnu by nie było fakapu")
- nie ma skonfigurowanych narzędzi, które wyłapywałyby brak pewnych rozwiązań (automatyczne scany w pipeline, np. Semgrep)
- brakuje wiedzy o tym, na co zwrócić uwagę
- ważniejsze od jakości testów jest pokrycie kodu testami
- ...itd.

Dlaczemu?

- nie usuwamy komentarzy / innych danych,które nie powinny zostać w aplikacji (ale czemu mają być w pierwszym miejscu...)
- błędna konfiguracja (aplikacja, infrastruktura)
- niewłaściwe zaprojektowanie aplikacji (np. zwracamy użytkownikowi kody błędu)

(https://portswigger.net/web-security/information-disclosure)

Co może pójść nie tak? O_O

Wiadomości o błędzie

- użytkownik może wpisać inną wartość, niż byśmy się spodziewali (np. zamiast numerycznej wartości, tekst)
- każdy język programowania ma swoje typy danych
- najważniejsze proste typy: string (tekst), int (liczba całkowita), float (liczba zmiennoprzecinkowa), boolean (prawda / fałsz) etc.

Wiadomości o błędzie

- walidacja po stronie serwera - ochrona przed atakującym
- walidacja na frontendzie - wskazanie dla użytkownika
- obsługa błędów - kluczowa! (np. try - catch)

Wiadomości o błędzie - Lab

https://portswigger.net/web-security/information-disclosure/exploiting/lab-infoleak-in-error-messages

Komentarze w kodzie

- zazwyczaj - pomagają zrozumieć szybciej kod
- nie wszystkie jednak powinny znaleźć się w repozytorium...
- ... a tym bardziej na produkcji!

Komentarze w kodzie - Lab

https://portswigger.net/web-security/information-disclosure/exploiting/lab-infoleak-on-debug-page

Kopia zapasowa

- strategia 3-2-1
- 3 kopie / 2 lokalnie na różnych nośnikach / 1 zdalnie (np. w chmurze)

(https://experience.dropbox.com/pl-pl/resources/3-2-1-backup-strategy)

Kopia zapasowa

- warto zabezpieczyć hasłem...
- ... i trzymać w rzeczywiście bezpiecznym miejscu.
- problem: włączone listowanie zawartości katalogów na serwerze
- problem - ogólny dostęp do katalogów

Kopia zapasowa - Lab

https://portswigger.net/web-security/information-disclosure/exploiting/lab-infoleak-via-backup-files

Dostęp przez localhost

- wydaje się być dobrym pomysłem - w końcu lokalnie aplikację odpala tylko ktoś, kto ma repozytorium, prawda?
- a co jeśli da się "podszyć" pod localhost?

Nagłówki HTTP

- przesyłają dodatkowe informacje w zapytaniu i / lub w odpowiedzi

(https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers)

Dostęp przez localhost - Lab

https://portswigger.net/web-security/information-disclosure/exploiting/lab-infoleak-authentication-bypass