Turvaline programmeerimine

- Miks programmid ei ole turvalised?
- Üldprintsiipe
- Avatud lähtekood ja turvalisus
- Sisendi kontroll
- Väljund, ründekindlus
- Enimohustatud programmiliigid
- Sisendi valideerimine
- Võidujooksud
- Puhvri ületäitumine
- Süstimisründed
- Muu

Miks programmid ei ole turvalised?

- Programmeerijaid ei õpetata piisavalt
- Ei mõelda mitmekasutajasüsteemidele
- Programmeerijad on laisad
- Programmeerijad on kehvad programmeerijad
- Programmeerijad pole turvaspetsialistid
- Turvalisuse tagamine võtab aega ja raha
- C/C++ on ebaturvalised keeled
- Kasutajad ei hooli
- Turvamudelid on kehvad
- Ei kasutata formaalset verifitseerimist
- Palju vana katkist tarkvara on kasutusel

Üldprintsiipe

- Turvalisus tuleb algusest peale sisse disainida, mitte hiljem paigata
- Turvalisust saab kontrollida mingil konkreetsel ajahetkel kehtiva seisundi kohta
- Ei saa teha nimekirja keelatud tegevustest
- Paranoia on programmeerimisel vooruseks
- Keerukus on vaenlane

Avatud lähtekood ja turvalisus

- Kas avatud lähtekood annab turvalisust juurde?
- Rohkem ülevaatajaid $\Rightarrow$ rohkem vigade leidjaid
- Võimalus veenduda koodi kvaliteedis
- Avatud kood iseenesest ei garanteeri kvaliteeti
- Ründajal on rohkem infot
- Ründajal on vähem eeliseid kaitsja ees
- Trooja hobused — kust tulevad?
- Parandamise võimalus
- Kokkuvõtteks: avatud lähtekoodil on parem potentsiaal *saada* turvaliseks

Sisendi kontroll

- Programm peab edukalt toime tulema igasuguse sisendiga, mida talle võidakse anda, ka vigasega
- Aktsepteeritav sisend peab vastama soovitud mustrile
- Aktsepteeritav sisend peab vastama soovitud mahupiirangutele
- Sisendiks pole ainult sisseloetavad andmed — ka süsteemist saadu
- Kontrollida protsessoriaja, mälu, I/O, võrgu kasutust (DoS oht!)
- Testida ka valede läbilaskmiste ja valede filtreerimiste vastu (mitte ainult oodatud tulemuste kohta!)
- Sisendit kontrollida iga mooduli liidese juures, mitte ainult kasutajaliideses
- Lubatud mustrid vs keelatud mustrid?

Sisendi valideerimine

- Programmi argumendid (käsurida)
- Keskkonnamuutujad
- Failipidemed
- Failide sisu
- Veebis: URLi kodeering (+UNICODE!!!)
- Veebis: küpsised (*cookies*)
- Veebi puhul ei piisa kliendipoolsest andmete valideerimisest (JavaScript, . . .)
- Signaalid
- `umask`
- Jooksev kataloog
- . . .

Väljund, ründekindlus

- Konfidentsiaalset infot mitte väljastada
- Ootamatu väljund, veateated!
  - Näide: veateated veebis
- Ründekindlus: et ühe kaitse läbimine ei annaks veel võitu
- Kas keskkond on õige?
- Kas andmed on kooskõlalised?
- Kas programmi ennast on modifitseeritud?

Võidujooksud

- Võidujooks — programmi töö korrektsus sõltub ajast (võistlusest välise teguriga)
- Tekib jagatud ressursside sünkroniseerimata kasutamisel
- Võidujookse võib tekkida nii usaldatavate kui mitteusaldatavate protsessidega
- Näide: ajutise faili varastamine
- Näide: nimeviidete kaudu muude failide üle kirjutamine
- Juhuslikud failinimed — kas ikka juhuslikud?
- Lahenduseks on atomaarsete operatsioonide kasutamine
- Lukustamine — vahend sünkroniseerimiseks, enamasti sõbralike osapoolte vahel

Puhvri ületäitumine

- Puhvri ületäitumine (*buffer overflow*) — üks levinumaid turvaprobleeme programmides
- Juba Morrise Interneti-uss kasutas sellist auku aastal 1988
- Von Neumanni arhitektuur: programm ja andmed on samamoodi samasuguses mälus
- Lohakas programmeerija ei kontrolli, kas andmed reaalselt puhvrisse mahuvad
- Kirjutatakse üle mälu puhvri järelt
- Pinus asuvate puhvrite puhul soditakse ära pinu ja muudetakse tagasipöördumisaadress
- Ka 1-baidist puhvri ületäitumist on turvaauguna ära kasutatud!

SQL süstimine

- SQL — andmebaasipäringute keel, tihti kasutusel ka veebirakendustes
- Oma metamärgid, mis tuleb välja filtreerida
- SQL manipuleerimise näide:
  `"SELECT password FROM people`
  `WHERE name='"+$name+"'"`
  Mis saab, kui
  $name=`"a'; DROP TABLE people; --"` ?
- `"SELECT FROM people where ID="+$ID`
  `$ID="3 OR 1=1"`
- Ohtlikke märke: ' " / \ * # % & () , : ; |
  . . . ja mitte ainult – lubage ainult tähti ja numbreid, kui võimalik

Muud süstimisründed

- SQL oli manipuleeritav, kuna struktuurne tekstipäring pandi valideerimata juppidest kokku
- Analoogselt on rünnatavad muud tekstipõhiste päringustringidega keeled
- Näiteks
  - XPath/XQuery

    ```
    /orders/client[@id='] | /* |
    /foo[bar='']/order/item[price >= 10]
    ```

  - LDAP otsifiltrid
  - Opsüsteemi käsuread

Veel

- Kontrollige kõigi funktsioonide poolt tagastatavat tulemust vigade suhtes
- Kasutage väliseid komponente ainult programmeerijale mõeldud liideste kaudu
- Ärge kasutage varieeruva tähendusega ega ilmselt ebaturvalisi teegifunktsioone
- Käsitlege erilise ettevaatusega mitteusaldatavatest allikatest pärit edasisi viiteid (*"Web Bugs"*)
- Peitke salajast informatsiooni (paroole logis, võtmeid mälus, andmeid URLis)
- Ärge asjata tehke `setuid`/`setgid` programme; kui teete, siis pisikesi

CWE haavatavuste top 25

- SQL süstimine
- OS käsu süstimine
- Puhvri ületäitumine
- *Cross-site scripting* (XSS)
- Puuduv autentimine kriitilises kohas
- Puuduv pääsukontroll
- Sisseprogrammeeritud juurdepääsuinfo
- Ohtlikku tüüpi failide piiramata üleslaadimine
- Ebausaldusväärse sisendi põhjal turvaotsuste tegemine
- Liigsete privileegidega programmi käitamine
- *Cross-Site Request Forgery* (CSRF)

- *Path Traversal*
- Koodi allalaadimine ilma tervikluse kontrollita
- Vigane pääsukontroll
- Funktsionaalsuse laadimine usaldamata osapoolelt
- Lõdvad juurdepääsuõigused
- Potentsiaalselt ohtliku funktsiooni kasutamine
- Murtud või kahtlase krüptoalgoritmi kasutamine
- Vigane puhvri suuruse arvutamine
- Vigase autentimise puhul kordade arvu piiramata jätmine
- URL ümber suunamine mitteusaldatud saiti
- Ründaja etteantav formaadistring
- Täisarvude ületäitumine
- Räsifunktsiooni kasutamine ilma soolata