Web applications
Security and scalability checklist

Cuprins:

1. Introducere

Bogdan Belu - www.bogdanbelu.ro

- 16 ani de experienta tehnica (administrare de sistem, programare, datacenter)

- din 2001 director general la Distinct (www.distinct.ro), furnizor de managed services si datacenter sub sloganul “plus prin responsabilitate”

Definirea problemei

Despre ce discutam? Checklist-ul unui administrator de sistem responsabil de sistemele pe care ruleaza o aplicatie web.

- doctrina de securitate se transpune intr-un plan de implementare

- estimare necesitati (SLA, targets) vs disponibilitati (buget)

- nu poti proteja 100%, obiectivul este sa faci neviabil atacul asupra sistemului

- principii: KISS, usable, multiple layers, de fond nu de forma, “expect the worst”, "security is as good as it's weakest link"

- in general ai de-a face cu doua mari clase de amenintari:

  A. compromiterea securitatii informatice (patrundere neautorizata, furt de date, modificari de date, etc)

  B. Denial of Service

2. Checklist generic

2.1 Backups

2.1 Backups - "shit happens. e bine sa ai optiunea sa dai timpul inapoi"

- ideal: posibilitate de restore a sistemului la orice punct din trecutul apropiat (zeci de zile)

- faci backup ?

- merge si restore ? ("The backup procedure works fine, but the restore is tricky! ")

- ce viteza de restore ai ?

- in cat timp trebuie facut restore ? (RTO - Recovery Time Objective)

- cate ore de modificari iti permiti sa pierzi ? - (RPO - Recovery Point Objective)

- folosesti si backup incremental ?

- monitorizezi backup-urile ?

2.1 Backups

- daca ai mai multe servere la care faci backup, restore-ul va fi consistent ?

- backup-urile la bazele de date/application data sunt consistente ?

- crezi ca RAID este backup ?

- backup-ul trebuie sa fie criptat ?

- backup-ul este transmis securizat ?

- backup-ul poate fi compromis daca sursa este compromisa ?

- backup-ul este stocat securizat ? (poate deveni vulnerabilitate chiar el)

- ai un sistem la indemana pe care poti sa faci restore de test ?

- sistemele de backup&restore sunt "usable"? (ex: preferi sa faci restore manual la cateva fisiere in loc sa folosesti tool-ul de restore complet)

2.2 Monitorizare

2.2 Monitorizare - "ca sa rezolvi o problema trebuie mai intai sa stii de ea"

- ai un sistem de monitorizare ?

- ai mecanisme de detectie implementate ?

- ai mecanisme de alertare catre cei interesati ? (ex: mail, sms, tichet de incident, etc)

- ai implementate verificari periodice standard (ex: disk space, numar procese, load average) ?

- monitorizezi integritatea datelor aplicatiei ?

- trebuie implementate verificari specifice aplicatiei ? (ex: nr de row-uri dintr-un anumit tabel nu trebuie sa depaseasca X)

2.2 Monitorizare

- ai loguri ?

- sunt logurile suficient de detaliate ?

- logurile sunt stocate si remote ?

- logurile remote pot fi compromise daca sistemul este compromis ?

- ai nevoie de loguri detaliate ale tranzactiilor pe HTTP (inclusiv POSTs, request headers) ?

- ai nevoie de un IDS ?

- ai tool-uri de securitate ce scaneaza sistemul periodic ? (ex: antivirus/security suite, nessus, logwatch, msec, rootkit detector)

- rapoartele de la tool-urile de securitate sunt verificate metodic de cineva ?

- ai nevoie de un file integrity checker ? (ex: samhain)

2.2 Monitorizare

- stii cum functioneaza sistemul in mod normal (ca referinta) ?

- ai grafice pe datele numerice returnate de senzori ? (ex: nagios perf data)

- graficele sunt/trebuie verificate periodic ?

- intervalul la care faci verificarile este suficient de scurt ?

- sistemul de monitorizare este suficient de usor de folosit (usability) ?

- numarul de false pozitives returnat de sistem este rezonabil ?

2.3 Detectie si rezolvare

2.3 Detectarea modalitatii de infiltrare si rezolvare - "cum iti dai seama prin ce modalitate a fost compromis sistemul"

- ai o procedura standard prin care investighezi bresele de securitate ?

- ai toate informatiile de care ai putea avea nevoie logate pe un sistem necompromis ?

- pui sistemul offline si/sau ii faci o copie in timpul investigatiei ?

- esti atent sa nu distrugi dovezile in timpul cercetarii ?

- tii un log cu operatiile facute in timpul investigatiei ?

2.3 Detectie si rezolvare

- poti produce un quick fix rapid ? daca nu poti, cine poate sa te ajute si in cat timp/la ce costuri ?

- ai timp sa repari vulnerabilitatea inainte de a repune sistemul in productie sau trebuie sa lucrezi pe live ?

- e nevoie sa ai un sistem complet de rezerva pentru a-l inlocui pe cel compromis ?

2.4 Securitate retea

2.4 Securitatea schimburilor de date prin retea - "pentru ca sistemul tau interactioneaza cu altele"

- ai identificat fluxurile de date in/out ?

- pentru fiecare flow te-ai intrebat daca ai nevoie de:

  - confidentialitate
  - asigurare integritate
  - autentificare
  - autorizare
  - non-repudiere
  - audit

- ce se intampla daca un alt sistem cu care interactionezi dar de care nu esti responsabil este compromis si trimite date periculoase ?

2.5 Sistem de lucru

2.5 Sistem de lucru – este important sa lucrezi metodic

- ai implementat un sistem de tichete de incident ?

- ai un knowlege base cu documentatie ? (ex: instructiuni de instalare, cum stii ca sistemul functioneaza normal, date de contact persoane implicate)

- informatiile din knowledge base sunt accesibile doar persoanelor indreptatite ?

- ai definite niste proceduri de lucru generale ? (ex: OS upgrade)

- ai definite niste proceduri specifice de lucru pe aplicatia web respectiva ? (ex: cum se da restart la firewall fara sa afectezi functionalitatea aplicatiei)

- daca apar schimbari in doctrina le poti implementa usor ?

- sistemul de lucru este usable ? (altfel nu va fi folosit cum trebuie)

2.5 Sistem de lucru

- cand esti obligat sa accepti un compromis de securitate (ex: ridicare restrictie per IP pentru ca cineva trebuie sa acceseze urgent de acasa) evaluezi riscurile si informezi factorii de decizie ?

- schimbarile in sistem sunt documentate si verificate la fel ca setup-ul initial ?

- schimbarile temporare devin permanente ? cum previi acest lucru ?

- documentezi schimbarile ? (ex: tichete, knowledge base, comment-uri in config files)

- informezi de schimbari si celelalte parti implicate ?

- sunt alocate suficente resurse pentru rezolvarea fluxului de tichete create ?

- ai o adresa de abuse@domain.tld pe care sa primesti reclamatii/sesizari ?

- sistemul de lucru este el insusi secure ?

2.6 Scalabilitate

2.6 Scalabilitate - cresti capacitatea sistemului in caz de DoS

- ce faci in caz de DoS ?

- poti opri atacul la sursa ?

- poti opri atacul macar partial (ex: blocare link de net international) ?

- ai mecanisme de detectie si blocare ? sunt aceste mecanisme scalabile in cazul unui atac de amploare ?

- ce metode de accelerare a codului ai la dispozitie (ex: apc, hphp, eaccelerator, flashcache, reverse caching proxy) ?

2.6 Scalabilitate

- implementezi de la inceput sau poti implementa rapid metode de scalare orizontale ?

  - frontends/reverse proxies (ex: varnish, nginx)

  - mai multe application servers (ex: Apache)

  - file system distribuit

  - object storage distribuit (ex: memcache)

  - database distribuit (ex: replication, clustering)

- poti modifica repede in fisierele de configuratie ale aplicatiei ? (ex: sa schimbi datele de conectare la sql)

3. Checklist specific

3.1 Virtualizare

- folosesti masini virtuale ?

- folosesti clonare de masini virtuale ? (ex: clona devel, scalabilitate, templates)

- folosesti live migration ?

- separi serviciile in masini virtuale separate ? (ex: sql, reverse proxy, web, phpbb)

- nu rulezi aplicatii in masina fizica ?

3.2 System updates

- in general, poti sa faci updates fara sa afectezi permanent functionalitatea aplicatiei web ?

- faci system updates ?

- update-urile sunt automate sau manuale ?

- procesul de update este supervizat de cineva ?

- cat de usor poti sa faci upgrade la kernel fara sa afectezi sistemul ?

- daca faci update la librarii strici aplicatia ? (ex: glibc, binutils)

- ai o procedura prin care sa faci update la aplicatiile generice folosite ? (ex: webserver, sql server, reverse proxy) ?

- poti sa faci update la php fara sa strici aplicatia web ?

3.3 Web application code

3.3 Web application code

- este nevoie de 'full code audit' sau de o evaluare de suprafata ?

- pui intrebari cheie developerului aplicatiei ?

  - aplicatia a fost scrisa respectand standarde de securitate ?
  - valideaza input-ul de la utilizator ? (POST, GET, Headere, Cookies, etc)
  - crede ca aplicatia este 100% sigura ? (daca e sigur e clar ca e o problema)
  - foloseste plugin-uri fara reputatie/neauditate/cunoscute cu probleme de securitate ?
  - foloseste CAPCHA acolo unde este cazul ?
  - datele sunt stocate securizat ? (ex: este nevoie de o partitie criptata?)
  - exista parole stocate in clar ?
  - aplicatia da prea multe informatii despre servere ? (ex: local paths, php notices, debugs)

3.3 Web application code

- aplicatia genereaza emailuri ? acestea pot fi folosite pentru spam ?
- se foloseste un sistem de source control ?
- exista procedura de instalare si descrierea aplicatiei in scris ? (peste 3 ani s-ar putea sa fie de folos)
- de ce versiuni de software din sistem depinde aplicatia ? (ex: PHP 5.2)
- foloseste rate limiting ca protectie la atacuri bruteforce (ex: contul de admin se blocheaza automat dupa 3 incercari nereusite de acces ?)

- zona de administrare a aplicatiei este protejata ? este nevoie de o protectie suplimentara ?

- codul aplicatiei este separat de datele aplicatiei ?

- cum se face deployment ? (ex: modificare direct pe sistemul live, modificare/testare pe devel)

- clona de devel este secure ?

- procedura de code audit se aplica si pe schimbarile ulterioare din aplicatie ?

3.4 Control acces

- controlezi toate portile de intrare in sistem ? (ex: nu cumva si-a mai facut developerul un shell pentru uz propriu)

- exista o politica de folosire a parolelor/cheilor de intrare in sistem ?

  - parolele sunt suficent de grele ?
  - parolele folosite uzual sunt prea grele ca sa fie tinute minte si trebuie notate pe ceva ? (le face mai putin sigure)
  - sunt preferate cheile in locul parolelor ?
  - aceasi parola este folosita pe mai multe servere ?
  - folosesti aceasi parola si pentru root si pentru mysql root ?
  - aplicatia web se conecteaza la baza de date ca root ?
  - parolele trebuie schimbate periodic sau la aparitia unui eveniment (ex: plecare angajat) ?

- exista accese folosite in comun de mai multi utilizatori ? (ex: toti developerii se logheaza ca admin)

3.4 Control acces

- poti pune la dispozitia developerilor/content managerilor un sistem de vpn usable (ex: ipsec, openvpn)

- este preferat sftp in locul ftp ?

- developerii/content managerii au IP-uri fixe sau dinamice ?

- suplimentezi controalele de acces din aplicatie (ex: admin login) cu protectiile tale 'known good' (ex: htaccess, restrictii pe IP) ?

- esti pregatit si pentru atacurile ce vin din interiorul retelei ?

- ai audit trail suficent de detaliat pe control acces in loguri ?

- ai evaluat si controlul accesului fizic la echipamente si linii de transmisie ?

- esti familiar cu tehnicile de social engineering si te protejezi de ele ?

3.4 Control acces

- instalezi firewall si pe fiecare componenta a sistemului ?

- politica default de firewall este REJECT/DROP ?

- controlezi din firewall atat intrarea pachetelor in sistem cat si iesirea pachetelor inspre exterior ? (ex: accesul pe port 80 dinspre server spre internet)

- firewallul este DNS-aware ? (se pot pune reguli si pe hostname in cazul ip-urile dinamice sau site-urilor din exterior ce isi schimba ip-ul des)

- exista portite de comunicare de informatii spre exterior dinspre server (ex: via icmp echo request sau dns requests ?

- poti pune in firewall reguli ce expira automat dupa un anumit timp ?

3.5 System hardening

- dupa instalare aplici metodele de reducere a "suprafetei de risc" a sistemului ? (ex: scos parole slabe, sters useri in plus, shells, suids, servicii/procese nefolosite, etc) ?

- aplici metodele standard de securizare pentru aplicatii ? (ex: Apache, PHP)

- aplici metodele standard de securizare pentru aplicatii web stock ? (ex: Wordpress, phpBB, Joomla)

- faci un pentest pe versiunea finala a sistemului ?

- faci un scalability test pe versiunea finala a sistemului ?

- folosesti security policies ? (ex: SELinux)

3.5 System hardening

- separi privilegiile pentru utilizatori si aplicatii ? (ex: cron-urile din aplicatia web nu trebuie sa ruleze ca root)

- folosesti rate-limiting contra brute force pe diversele componente ale solutiei ?

- instalezi firewall ?

- ai nevoie de IPS ?

- ai nevoie de un IDS ?

3.6 Toolbox

- ai un 'toolbox' de diagnoza si analiza ? (ex: live rescue cd, tcpdump, ngrep, wireshark, dstat, atop, foremost)

- esti familar aceste tool-uri ?

- aceste tool-uri sunt/trebuie instalate pe sistem de la inceput ?

- daca nu sunt instalate deja, se pot instala repede la nevoie ?

3.7 Informare permanenta

- ai mecanisme prin care sa fii informat de noutatile (vulnerabilitati, updates) legate de software-ul pe care il folosesti ?

- informarile critice de le vendori ajung direct in sistemul de tichete ?

- transmiti valorile de securitate in care crezi si celorlalti implicati in functionarea aplicatiei ? (programatori, content manageri, project manageri)

- atunci cand ti se cere sa faci compromisuri pe securitate comunici corect impactul estimat ?

- iti dezvolti permanent cultura profesionala ?

Concluzii

Doctrina ta de securitate:

1) sa existe si sa includa cel putin elementele de baza

2) sa fie implementata intr-un sistem sustenabil in timp

3) sistemul nu trebuie sa te lase sa uiti sau sa treci cu vederea principiile de la care ai plecat

4) sistemul trebuie sa includa si o componenta de actualizare periodica pentru el insusi

Incheiere

- Aceasta prezentare o gasiti si online la www.bogdanbelu.ro

- Q&A