Session 5   / /

بسم الله نبدأ .. رب اشرح لى صدرى ويسر لى أمرى.

Chapter 8

ليه اصلاً عندى فكرة ال users and groups ؟

- اول حاجه اصدر ال ownership بتاعت ال system resources
- تانى حاجه اعمل permissions لل resources دى أو لل users نفسهم يعنى مثلاً كل file ليه permission وكذلك ال processes نفسها ليها permission .. ليه طيب عشان هى اللى هتعمل مثلاً open و read و write وهكذا لل files دى

طيب عايزين نعرف أهم ال files عندنا ..

① /etc/passwd

دة بيحتوى على ايه ؟

- login name : - unique
- encrypted password : - empty means no password

بعد الكلام دة عندى مثلاً لو فتحت ال /etc/passwd هلاقى منظر زى دة مثلاً ؛

root : x : 0 : 0 : root : /root : /bin/bash

هنا دى لو فاضيه معناه مفيش password طيب لو فيها char زى دى كدة ؟ ال password هيكون موجود بس encrypted فى ال shadow file

- User ID (UID) : - 32 bit , 0 → root

- Group ID (GID) : - 32 bit

- Comment : - extra info about the user

- login Shell :

ال Shell اللي بعمل منها login ولو فاضية هبدأ من /bin/sh هي اللي هتكون ال Shell بتاعتي وهي أول Shell بتفتح لما ال user يعمل ال login ال login shell دي

- home directory :

كل process ليها cwd بتاعها ودة ال directory اللي بحسب من عنده ال relative path طب لما ال user يعمل login دة المكان اللي ال User يعمل login هيبدأ ال login shell تشتغل عنده

طب بيقى أنا عندي في home directory ال file اللي موجود عندي وفيه الحاجات بتاعته ووصلها ال login shell المشتغل ودة بيكون ظاهر عندي في ال prompt اللي هو الجزء بتاع ~ : $ دة المفروض أنا ب refer لل home بتاع ال user اللي أنا عنده عشان كدة لو طلعت برا خالص عند ال root هلاقيه هو نفسه file برا أصلاً جواه ال Users اللي عندي وجواه حاجات ال user دة وهو دة المكان اللي ال login shell بـ operate منه زيها زي أي process

* ليه نعمل users لحاجات زي man , apt ؟
كل الحاجات دي في الاخر هي بتشغل processes ، فالتال أنا عايزة يكون عندي control للحاجات اللي هي بتعملها ف أنا ممكن اشغلها ب user معين عشان اديها permissions معينة

يعني مثلاً لو معمول لل man user ف أنا هعمل ال man program بيقرأ مثلاً ال man pages of بس مش بيعدل فيها ف هعمله user وال user دة هيكون no login user هو دة اللي هيقوم ال man process وهيكون no login

* /etc/group

كا ال unix افتل كاكل user ة له جروب واحد بس عشان كدة في etc/passwd واحد بس GID وبعدين قالوا لو عايزين ال user يكون موجود في كذا جروب ف قالوا هنعمل file اسمه etc/group وده معرفة ال files اللي موجودة في ال system وده فيه

- group name : - unique
- encrypted passwd : زي ال etc/passwd بس هي نادراً ما بستخدم ودي موجودة في /etc /shadow
- group ID
- user list

*/etc/shadow

ده هو يحل مشكلة ال security عشان بعد ما كانت ال Password موجودة encrypted في ال etc passwd
واى حد يقدر يشوفها وممكن نعمليها decryption ونبقى
كدة سهل اعمل اى حاجه قالوا لا احنا نحطها في
file تانى محدش يقدر يفتحة غير ال root
ودة هو

- login name : unique , matching name in etc /passwd

- encrypted password

- aging information :
شوية معلومات عن ال password ممكن نجيبها من ال man

طب السؤال المهم ازاى ال password بيحصلها encryption

عشان نعرف دة لازم نعرف ايه هى ال hash function ؟

هى زى mathmatical function أو algorithm بتاخد
Variable number of characters "message"
وبتحولها ل fixed length string
اسمها hash

طب هو بمنتهى السهوله ممكن يحصل Collision
مثلاً لو ال fixed length string عبارة عن 2 bytes
ف أنا مثلاً عندي output يساوي 2^{16} احتمال
لكن ال input ليه ∞ احتمال

ف بالتالى ممكن two different strings يطلعولى نفس ال Hash وده اسمه Collision وكل ما يكون احتماليته اقل كل ما يكون افضل

ال Hashing دى عليه one way يعنى مينفعش ارجع ال Hash لـ string تانى

Hash function properties :-

① Pre-image resistance : means it's one way

② Collision resistance :
مينفعش أو لازم تقل احتماليه أن two messages يطلعوا نفس ال hash

الى فيها الخصائص دى بتسمى Crypto graphic hash function

طب ما هو ال Authentication : هى عمليه انى اعرف مين ال user اللى داخل ال system عندى وغالبا بيتعمل بـ user name & password

طب أزاى يعرف ال password بتاعت ال user انها هى الصح ؟
مش الصح طبعا انى اسيفها على ال system عندى والا اى حد هيعمل login هيعرف ال password دى
ف أنا مش هسيفها ولا encrypted value بتاعتها عشان ممكن يتعملها decryption ف أنا هسيف ال hash بتاع ال password دى ولما اجى ادخل أنا ال password هعمل عليها hashing و أقارن ال hash بال hash

طب ايه الفرق بين ال Hashing وال encryption ؟

الكلام دة بجد شوية بحث
أول حاجة ال encryption :
هي عملية تشفير ل data معينة محدش غير اللي يقدر يفك
التشفير دة يقدر يقراها .
ودة ليه نوعين Symmetric ، Asymmetric

أول حاجة ال Symmetric دة بيشفر ال data وبيكون ليها key
بيفك بيه التشفير دة طب وال key ؟ أروح أشفره
والأفضل كدة في Closed loop يفضل عشان ال data
تفضل Safe
ف عشان كدة اتعمل ال Asymmetric
مثلا لو عندي mail box في الشارع لو عايزه ابعت
هحط فيه ال mail دة طب لو عايزه اقرأ اللي فيه ؟
لازم يكون معايا مفتاحه . ف عشان كدة اخترعنا فكرة
ال RSA Algorithm
ودة مبني على فكرة ان يكون فيه ليها مفتاحين واحد public
وواحد private ال public دة مع اي حد ودة بستخدمه
في ال encryption يعني مثلا لو عندي message عايزه
اعملها encryption مستخدم ال public key دة
وال private هو اللي يقدر يعمل decryption
ف بالتالي لو عايزه اعمل encryption ل data هاخد
ال public key بتاع الشخص اللي هيتبعتله ال data
واروح اعملها encryption وبعتها حتى أنا نفسي معرفش
اعملها decryption طالما مش معايا ال private key

/ /

- أزاى بقى البرامج بتعمل الموضوع ده؟
من library routine تانية اسمها crypt lib ودى فيها
ال encryption methods افتح ال man بتاعتها
هى بتاخد ال password وشوية settings كدة تعرفني
استخدم انهى hashing function

ف لو عايزة ا check دى password هستخدم ال crypt
دى طيب أنا بديله وأنا بـ login ال password بس
هو يعرف منين ال settings ولا ال hash function
دى؟ بيكون فى prefix كدة فى أول ال password
بيعرف منه انا استخدمت انهى hash function بالتالى
ا hash بيها ال password واشوف هل الاتنين نفس
الحاجة ولا لأ.

فى شوية library routines كدة هنحاول نعرف عنهم شوية
حاجات ونبقى نقرأ ال man برضو
- getpwname ()
دى هتروح تفتح ال /etc/passwd وتجيب ال entry
بتاعت ال user دة منه

- getspname ()
دى هتاخد معلومات ال user من ال /etc/shadow

- getpass()
دى بتخلى ال password متتكتبش لما اكتبها بـ أنى اقفل
ال echo بتاعت ال terminal ف مش هيعمل echo وقتها
وبعد ما اعمل enter نفتحها تانى

طيب لو أنا عملت program هيـ Check على الpassword
هو كدة من مواه لازم يفتح الـ shadow عشان ياخد منها
الـ hash بتاعت ويقارنها بـ الـ password اللي هيدهوله
بعد مايعملها hashing

طيب احنا كنا قولنا أن الـ shadow دة محدش يقدر يفتحه
غير الـ root فـ أزاي الـ program بتاعي هيا كسه؟

زمان كنا قولنا أن في حاجه اسمها الـ ID والـ effictive ID
الـ ID هو اللي الـ process بتـ inhert من الـ parent
على طول والـ effictive دة ID في الأول وغالبا هو نفسه
الـ ID الصاحب اللي باخده من الـ parent بس ميزة
أني اقدر اغيره عشان ادي الـ process بتاعتي permissions
تانية غير بتاعت الـ parent دي
فـ هروح اغيرها . طيب دة بيحصل أزاي
هعمل bit set_user_ID ودي لو عملتها
set هتشتغل بـ الـ permissions بتاعت الـ owner
بتاع الـ process دي لكن لو مش معمولها set هتاخد
الـ permissions بتاعت اللي شغل الـ process اللي عملها
exec اصلا"

memory allocation

أول حاجة أنا لو عملت program مثلاً بيعمل dynamic allocation
ونطبع الـ program break, &end, &edata, &text

أول حاجة نعمل لـ extern end, edata, etext
عشان اعرف أشوفهم

* أول ملاحظة مع كل execution هلاحظ الـ addresses بتختلف
كل مرة وهنقول ليه دة بيحصل.

* بس تاني ملاحظة هلاحظ الـ program break
بيتحرك في البداية خالص مع أننا كنا قولنا انه المفروض هو من
بداية البرنامج هيكون عند الـ &end ليه ده حصل؟
عشان ممكن الـ glibc تعمل هي malloc اصلاً لحاجات
حولها مش أنا لوحدي اللي بعمل malloc

← طيب لو أنا عايزة اعرف الـ virtual memory دي محولة
أزاي أو متقسمة ازاي هعمل ايه هي
/proc/'pid of process'/maps
موجودة في الـ file ده والـ ' ' دي هتروح
تنفذ الـ Command اللي فيها الاول بعدها تعمل replace
وتنفذ الـ Command كلة

1000 Hex = 4096 decimal

address - address + 1000 H → rwxp → offset → file name → تيكون شكله كدة مثلاً

لهنا بداية الـ Page

لهنا نهاية الـ Page

الـ Permissions

read - write - execute.

الـ offset بتاع الـ file اللي يقرأ منه

اسم الـ file او اسم المكان ده من الصيوري من مثلاً

1- شوية حاجات بيـ allocateها الـ glibc من بداية البرنامج خالص قبل حتى الـ text. بتاع الـ Program

2- text segment
ودي هلاحظ أن من الـ permissions بتاعته الـ execute

3- data segment

4- bss segment

5- heap

6- shared libs

7. stack

8-9.10 VVar , Vdso, Vsyscall

مش مهمين دلوقتي
بس ناخد نبذه عنهم

ال VVar هو Section المستخدمة في ال kernel عشان تحمل store
لل Variables per process اللي ليها علاقة بال Virtual memory

وال Vdso هو Virtual dynamic shared object الـ shared objects
دي بيستعملها mapping من الجزء ده وبطريقة ما بيتوفر عليا
ال overhead بتاع ال context switch لل kernel mode
هي فيها شوية system calls وال implementations بتاعتها
في المكان ده اللي ممكن اعليها في ال user space من غير ما اتحرك
للكامل لل kernel space.

طب ايه هي ال Vsyscall هي mechanism في ال Linux kernel
القديمة عشان ت optimize ال calling بتاع ال system calls
برضو بتوفر عليا ال overhead بتاع ال contexting بين ال user space
وال kernel space فبتعمل mapping لجزء صغير فيه برضو
implementations بتاعت شوية system calls كده توفر عليا الموضوع
ده الجديد بتاعها هو ال Vdso بس ليها اعتبارات
احسن في ال flexibility وال security اكتر من ال Vsyscall

طيب فى ملاحظة مهمة جداً، أنى لو عملت run هلاقى كل مرة الماكن بتتغير وده عكس مفهوم الـ Virtual memory اصلا الى هى مفروض من اصغر لـ FFFs عشان أول حاجة تتخلى كأن كل process ليها الميموري كلها وتانى حاجة تخلى البرنامج كل مرة يروح لنفس الاماكن وده من run

فليه ده بيحصل ؟

(ASLR) Address space layout Randomization فى حاجة اسمها

لما كانت الـ sections بتاعت البرنامج ثابتة فى مكان معين كان بيحصل حاجة اسمها الـ buffer overflow دى نوع من الـ attacks وطريقة ما بيخلى البرنامج يروح فى حتة فيها bug مثلاً أو اوصل لحتة فيها مثلاً ال password فدى طريقة تخلى اغير فى الـ behaviour بتاع البرنامج أو اوصل لـ sensitive data

فـ بالتالى افضل انى احط الـ sections بتاعت الـ program فى حتة مختلفة كل مرة بحيث ميقدرش الـ attacker يتنبأ بالـ address الى جاى لأى حاجة فى الـ program

ودى حاجة اقدر اعملها disable

(man proc → search for randomization)

طيب في ملاحظة مهمة جداً ، أني لو عملت run هلاقي كل مرة الاماكن بتتغير ودة عكس مفهوم ال Virtual memory اصلا اللي هي مهوية من اصفار لـ FFFs عشان أول حاجة تعلى كأن كل process ليها الميموري كلها وبأي حاجة تخلي البرنامج كل مرة يروح لنفس العنوان وـ run ومـ run
فـ ليه ده بيحصل ؟

في حاجة اسمها (ASLR) Address space layout Randomization

لو كانت الـ sections بتاعت البرنامج ثابتة في مكان معين كان بيحصل حاجة اسمها الـ buffer overflow دي نوع من الـ attacks لطريقة ما بيخلي البرنامج يروح في حته فيها bug مثلا أو اوصل لحته فيها مثلا الـ password فـ دي طريقة تخليني اغير في الـ behaviour بتاع البرنامج أو اوصل لـ sensitive data

فـ بالتالي الافضل اني احط الـ sections الـ program في حته مختلفة كل مرة بحيث ميقدرش الـ attacker يتنبأ بالـ address اللي جاي لأي حاجة في الـ program

ودي حاجة اقدر اعملها disable
(man proc → search for randomization)

طيب بعد ما اعلمها disable هلاقي ال addresses مش بادئه
من صفر مثلاً 8 هي بادئه من بعد كدة بكتير طب ليه الكلام ده؟

ده بسبب ال dynamic linking طب ايه الموضوع ده

احنا شوفنا من ال dynamic linking كما اصلاً اعمل Shared library
بعملها ب PIC f الى كان بيخليها Position Independent Code
ف بالتالي يكون لكل حاجه مكان كل مرة في الميموري وهي مش بتشتغل
بناءً على مكانها فين ... ف لما بعمل program بيطلع dynamic linked
هو كمان كأنه بيكون position independent و بالتالي الاماكن
مش هي نفسها اللي في ال elf بتاعه يعني ال map file
وال elf اماكنهم مش واحدة و بيرضو على حسب فهمي أنه
ال dynamic linker بياخد مكان من الاول يعني

ف لو خليت البرنامج بتاعي Statically linked هلاقي أن
الاماكن اللي في ال map file هي نفسها اللي في ال elf
طيب ما كنا قولنا أن ال Randomization هو اللي بيعمل الكلام
ده

الفكرة أن اللي هيثبت الحاجات اللي هتكون محتاجاها Statically
يعني مثلاً ال bss و data. و text. هيفضلوا مكانهم مش
هيتغيروا واللي هيتغير ال heap وال Stack مثلاً لو فتحت
ال Randomization تاني الحاجه اللي ممكن اغير مكانها بس
هي اللي هتتغير ف أنا محتاج ال dynamic linking.
وال Randomization الاتنين سوا عشان اغير مكان الحاجه
خالص بالكامل وتكون الدنيا Safe وميحصلش attacks