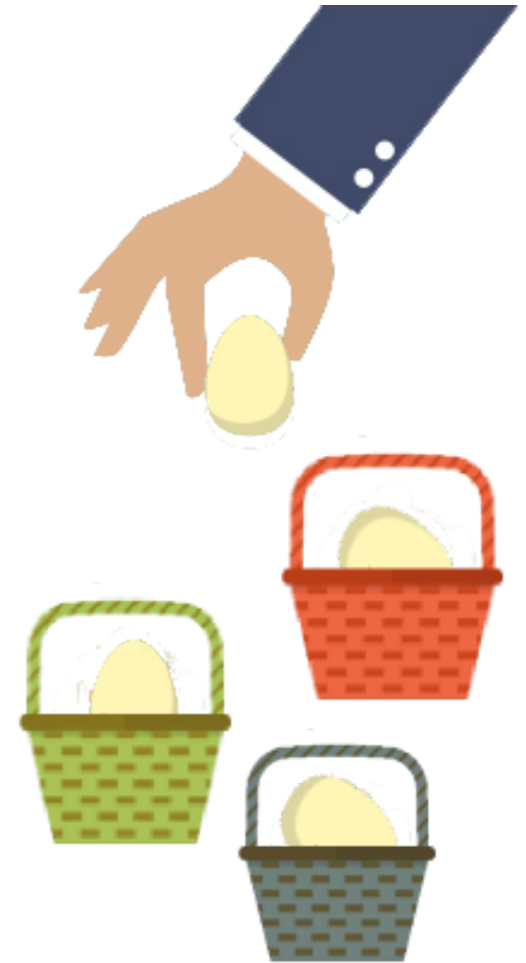


# Compartmentalization

Dividing our digital life to increase privacy



# Sharing is Caring

*Please copy, share, and remix!*

grab a copy of the presentation:

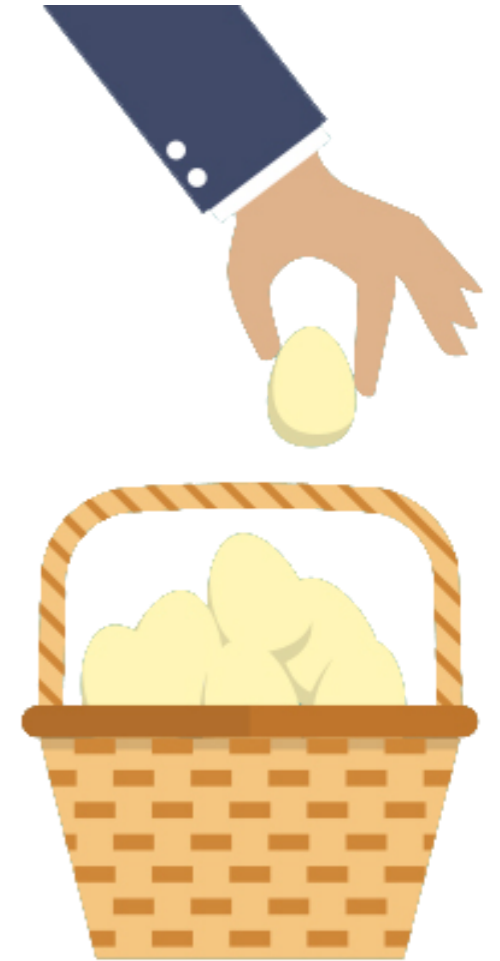
[github.com/francisco-core/presentation-compartmentalization](https://github.com/francisco-core/presentation-compartmentalization)

The file `presentation.md` includes additional notes

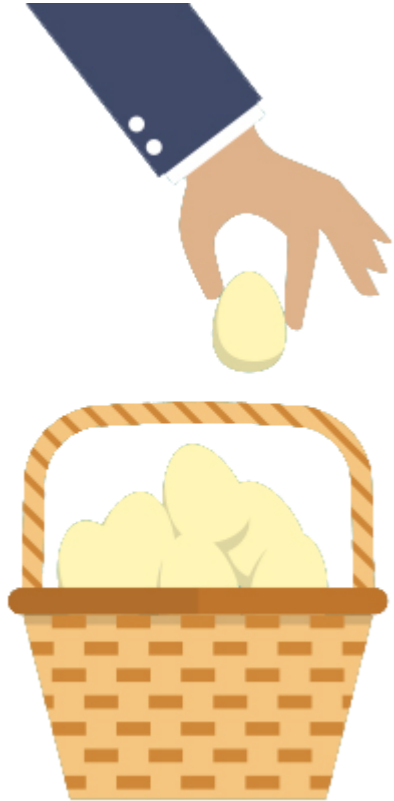


This presentation is under the Creative Commons Attribution-ShareAlike License

**Don't put all your eggs  
in one basket**



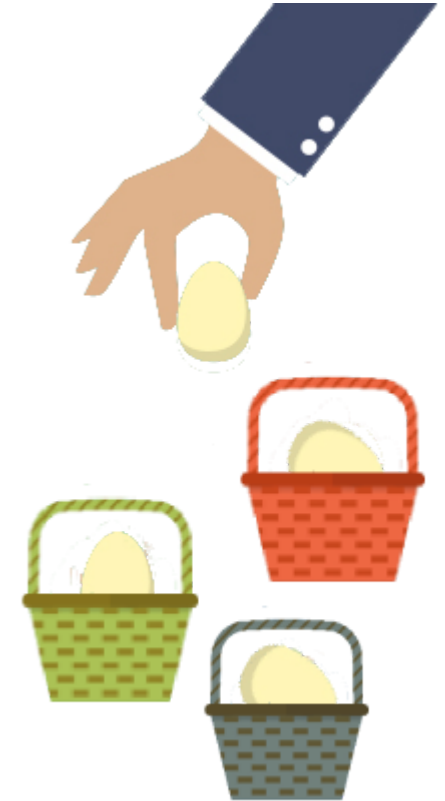
## centralized



Easier to guard

**Easier to break**

## compartmentalized



Harder to manage

**Tougher to break**

# How to compartmentalize?

1. Planing ahead
2. Practice what you planed

# Some terminology

to make things easier

# Domain

What we call each basket. As in *domain* of your life

E.g. Personal, Work, Banking, etc.

**tip:** Use colors to help you identify each one

# 1. Make a plan

Domain	color	Uses
<i>personal</i>	orange	visiting websites in perosonal account
<i>work</i>	turquoise	anything related to work
<i>banking</i>	green	for online banking only
<i>untrusted</i>	red	anything that does not fit the orther categories

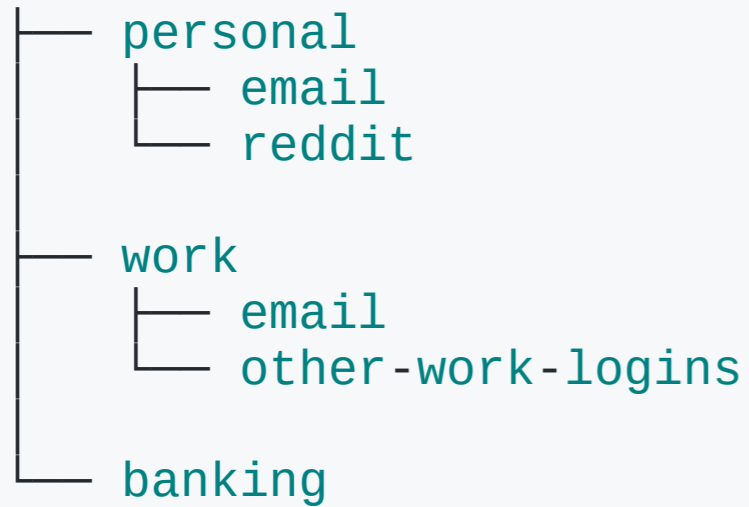
**tip:** make sure the colors have a meaning to you (e.g. green as safe)



## 2. Practice

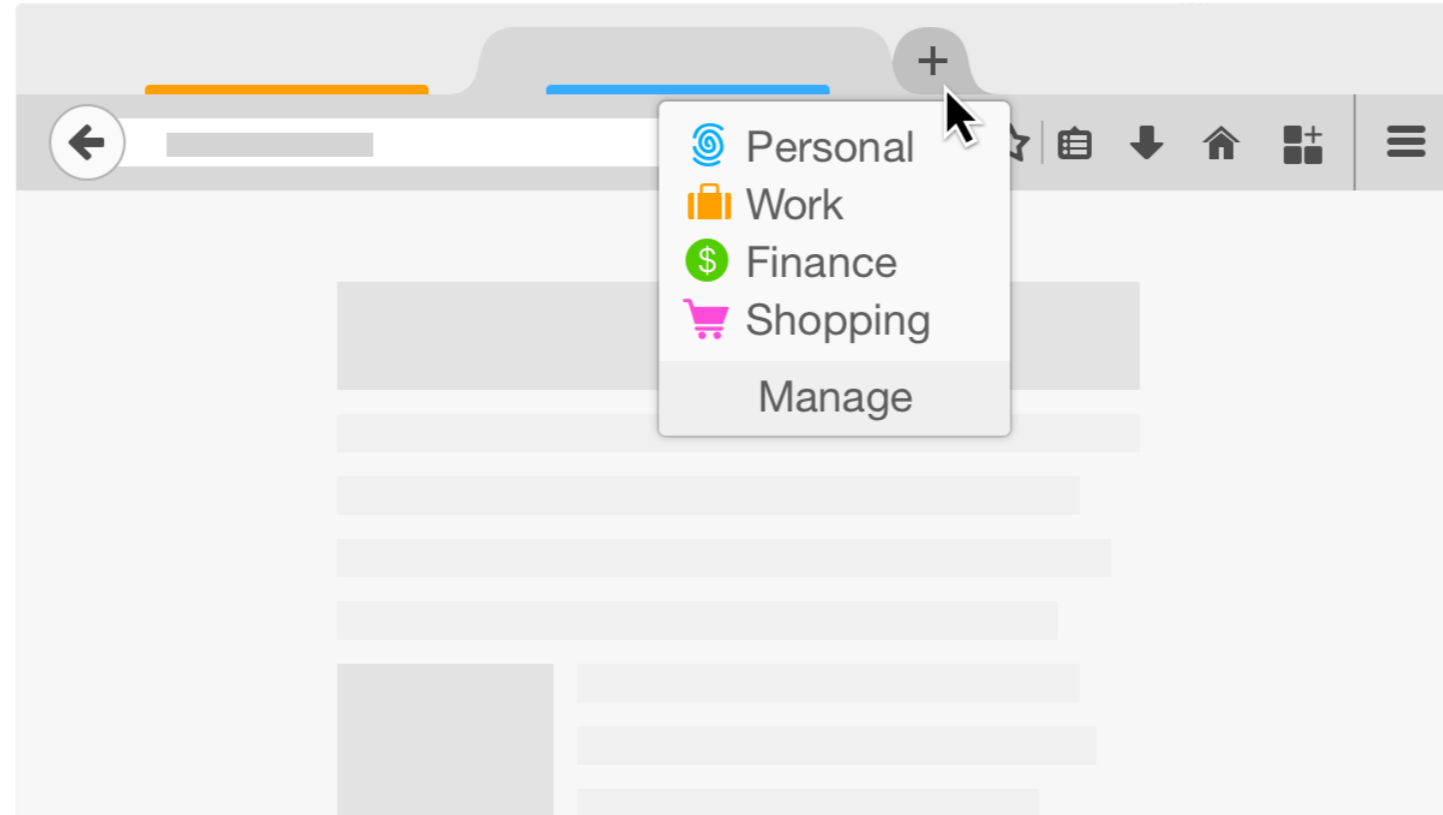
1. Different logins (with the help of your password manager)
2. Fake logins
3. Different service provider
4. Different browsers
5. Different phones (work and personal)

# Password Managers



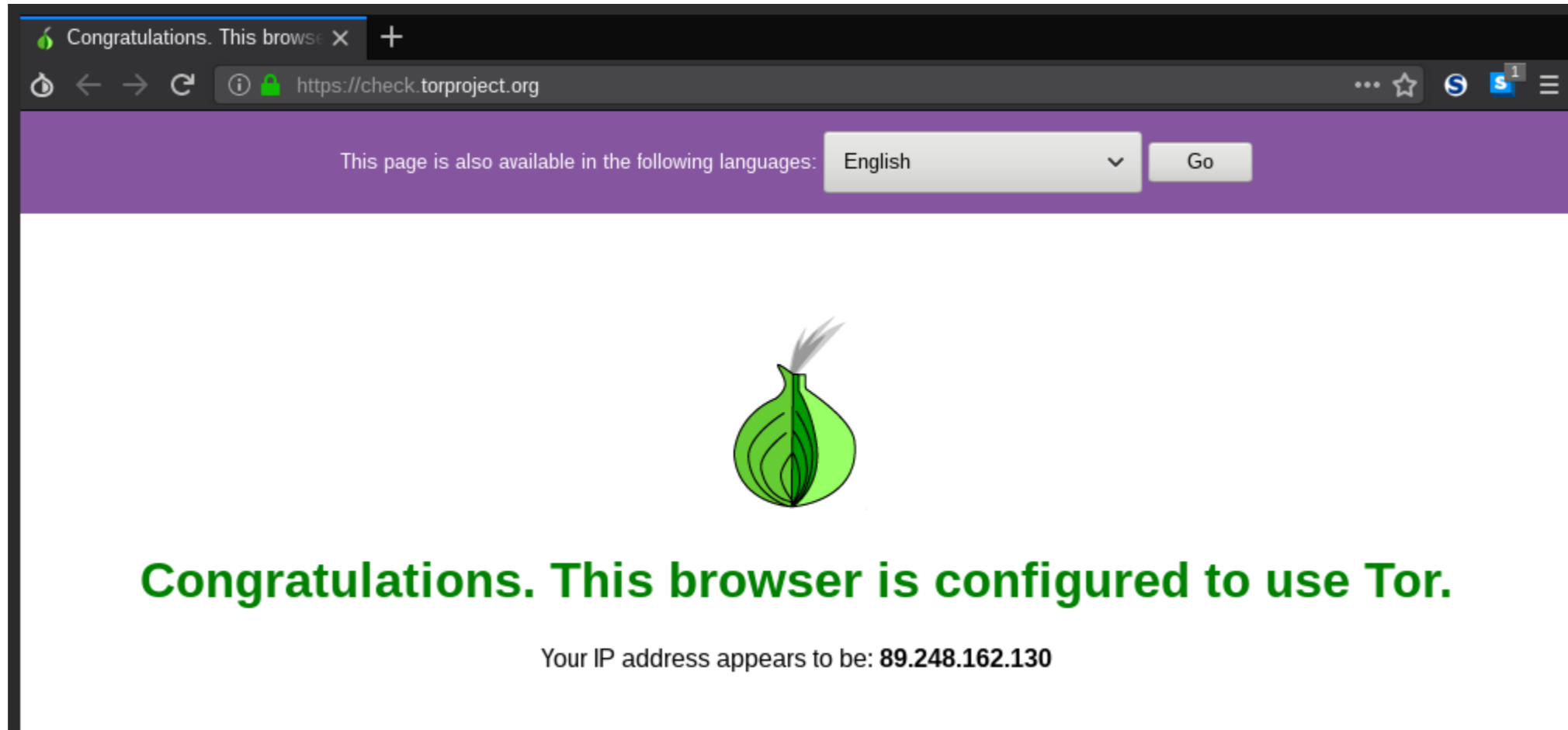
**Tip:** organization here is important. It's really easy to have a huge mess  
And with mess comes insecurity

# Firefox Containers - Demo



**warning:** firefox containers helps you privacy-wise in the sense that it allows you to manage various accounts at the same time. It **does not** protect against fingerprinting. For that, you should be using *Tor Browser*

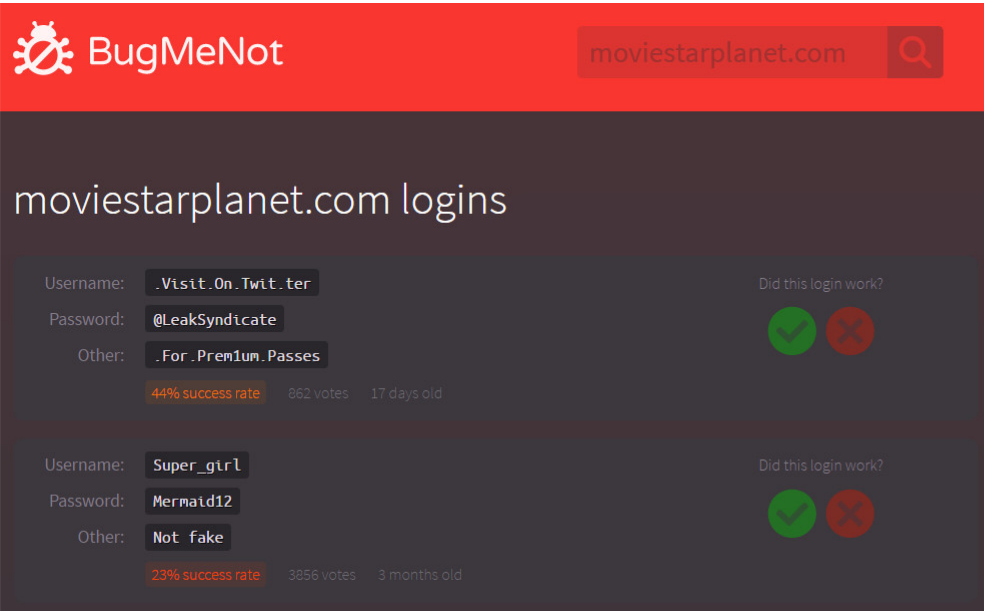

# Torbrowser - Demo



To switch identities you press the onion button in top left corner and it will restart your browser.

# Temporary Accounts

Not all eggs deserve a basket

 <p>The screenshot shows the BugMeNot interface for the website 'moviestarplanet.com'. It lists two login attempts. The first attempt has a success rate of 44%, 862 votes, and is 17 days old. The second attempt has a success rate of 23%, 3856 votes, and is 3 months old. Each attempt includes a 'Did this login work?' section with green and red checkmarks.</p>	 <p>The screenshot shows the Login 2 beta interface. It features a search bar with 'facebook.com' entered and a 'Get' button. Below the search bar, there are input fields for 'Login:' (containing 'loudacris_1115@yahoo.com') and 'Password:' (containing 'jamstel'). A 'Not working, show more' button is also visible. A tip at the bottom reads: 'Tip: for select login or password click on them'.</p>
<p>bugmenot.com</p>	<p>login2.me</p>

**Tip:** Only login with an account of yours when you really need to

# Various Computers



## QubesOS

**[varia] Software Guard Extensions Programming Reference**

ENCLAVE OPERATION

After AEX has completed, the logical processor is no longer in enclave mode and the exiting event is processed normally. Any new events that occur after the AEX has completed are treated as having occurred outside the enclave (e.g. a #PF in dispatching to an interrupt handler).

### 3.2.3 Resuming Execution after AEX

After system software has serviced the event that caused the logical processor to exit an enclave, the logical processor can re-start execution using ERESUME. ERESUME restores registers and returns control to where execution was interrupted.

If the cause of the exit was an exception or a fault, and was not resolved, the event will be triggered again if the enclave is re-entered using ERESUME. For example, if an enclave performs a divide by 0 operation, executing ERESUME will cause the enclave to attempt to re-execute the faulting instruction and result in another divide by 0 exception. In order to handle an exception that occurred inside the enclave, software can enter the enclave at a different location and invoke the exception handler within the enclave by executing the EENTER instruction. The exception handler within the enclave can attempt to resolve the faulting condition or simply return and indicate to software that the enclave should be terminated (e.g. using EEXIT).

#### 3.2.3.1 ERESUME Interaction

ERESUME restores registers depending on the mode of the enclave (32 or 64 bit).

- In 32-bit mode (IA32\_EFER.LMA = 0 || CS.L = 0), the low 32-bits of the legacy registers (EAX, EBX, ECX, EDX, ESP, EBP, ESI, EDI, EIP and EFLAGS) are restored from the thread's GPR area of the current SSA frame. Neither the upper 32-bits of the legacy registers nor the 64-bit registers (R8...R15) are loaded.
- In 64-bit mode (IA32\_EFER.LMA = 1 && CS.L = 1), all 64-bits of the general processor registers (RAX, RBX, RCK, RDX, RSP, RBP, RSI, RDI, RB, ...R15, RIP and RPLAQS) are loaded.

Extended features specified by SECS.ATTRIBUTES.XFRM are restored from the XSAVE area of the current SSA frame. The layout of the x87 area depends on the current values of IA32\_EFER.LMA and CS.L:

- IA32\_EFER.LMA = 0 || CS.L = 0
  - 32-bit load in the same format that XSAVE/FXSAVE uses with these values.
- IA32\_EFER.LMA = 1 && CS.L = 1
  - 64-bit load in the same format that XSAVE/FXSAVE uses with these values plus REX.W = 1

### 3.3 CALLING ENCLAVE PROCEDURES

#### 3.3.1 Calling Convention

In standard call conventions subroutine parameters are generally pushed onto the stack. The called routine, being aware of its own stack layout, knows how to find parameters based on compile-time-computable offsets from the SP or BP register (depending on runtime conventions used by the compiler).

Because of the stack switch when calling an enclave, stack-located parameters cannot be found in this manner. Entering the enclave requires a modified parameter passing convention.

For example, the caller might push parameters onto the untrusted stack and then pass a pointer to those parameters in RAX to the enclave software. The exact choice of calling conventions is up to the writer of the edge routines; be those routines hand-coded or compiler generated.

#### 3.3.2 Register Preservation

As with most systems, it is the responsibility of the callee to preserve all registers except that used for returning a value. This is consistent with conventional usage and tends to optimize the number of register save/restore operations.

34 Ref: # 329298-001

**[Dom0] Qubes VM Manager**

Name	State	NetVM	CPU Graph	MEM
dom0	Running	n/a		2598 MB
sys-net	Running	n/a		301 MB
sys-firewall	Running	sys-net		301 MB
varia	Running	sys-firewall		979 MB
work-web	Running	sys-firewall		1173 MB
work-mutt	Running	sys-firewall		604 MB
keys-itl-email	Running	---		478 MB
work	Running	sys-firewall		607 MB
personal	Running	sys-firewall		750 MB

[user@work ~]\$