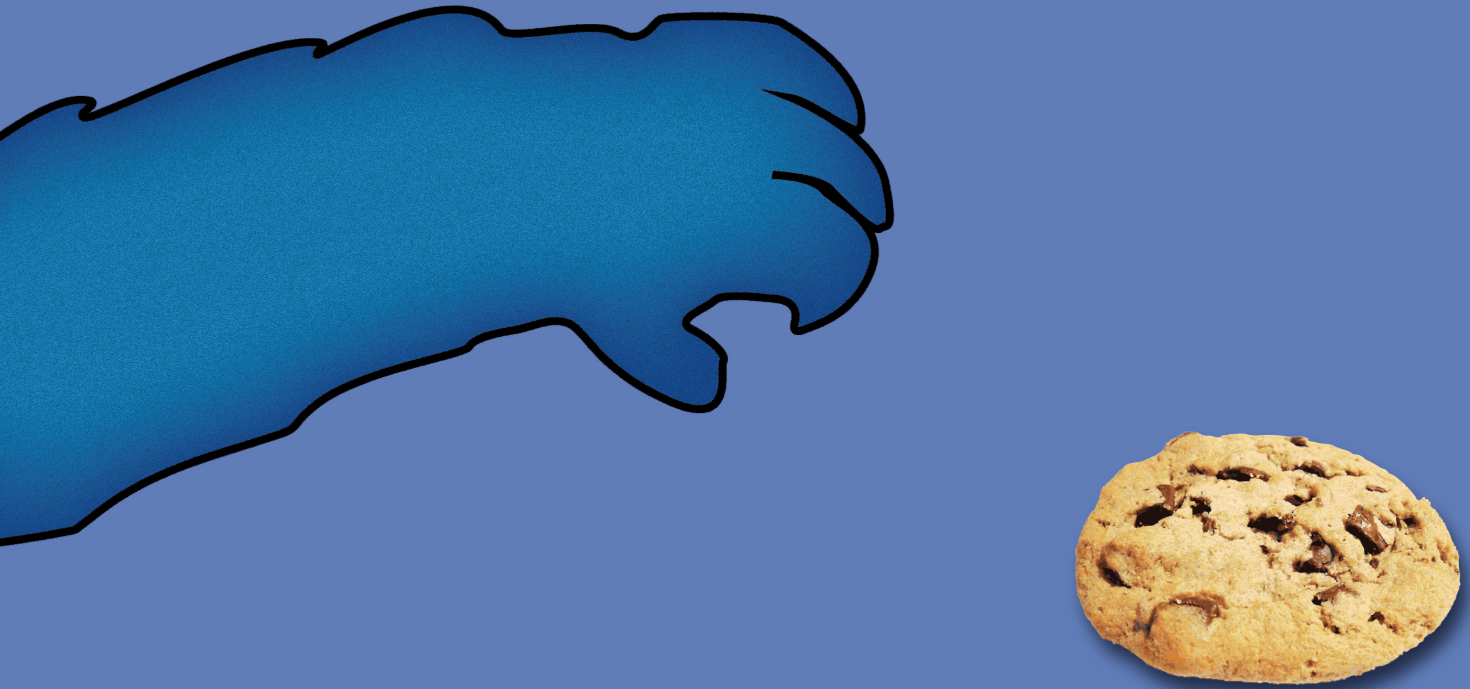
CookieMonster

Thomas Chopitea - Adrien Gsell
(Group 20)



CookieMonster

Thomas Chopitea - Adrien Gsell

Thursday, 19 may 2011

Abstract

Communication ubiquity is a paramount concern when it comes to the design and implementation of new technologies. The breakthrough brought by wireless technology such as Bluetooth and more importantly WiFi revolutionised the way we communicate - Internet can now be accessed from almost anywhere - your university campus, your office, your living room, the coffee shop down the street, the airport...

But this incredible increase in accessibility does not come without any downfalls - security in wireless networks has been a major concern since its earliest creation, and the test of time was everything but forgiving, exposing major holes in early WiFi encryption methods (such as WEP).

Security has been increased with WPA and WPA2 encryption methods. But this is far from being universally adopted, and lots of network are still weakly protected or just completely plain open.

The aim of this project is to focus on one the vulnerabilities that such lack of security renders trivial - cookie stealing (or cookie snarfing) on open wireless networks.

Open Wifi networks are everywhere

Fig. 1 of 5 Back to Album – CookieMonster's Profile Previous Next

Home: Open Wifi networks. Even though implementing security in wireless networks is quite easy, effectively securing a wireless network becomes increasingly difficult when it is intended to be used sporadically by a large number of users (e.g. coffee shops, bars, restaurants, or airports, conference centers...). Most solutions consist of open (unencrypted) wireless networks providing some kind of pay-as-you-go service. This is done automatically, and no configuration is needed client-side - which makes network setup really easy (and eventually expensive).

This might be really good for business, but what most users don't know is that all their data is being broadcasted from their wifi interface to everybody, and that someone with minimum technology and skills can read it and exploit it in ways the owner probably didn't intend to.

Cleartext passwords, instant messages, webpages requests, emails, everything can be intercepted and sniffed. This is one of the main vectors of attack for OWASP #3 issue, Broken Authentication and Session Management in web applications.

Info: HTTP Sessions, OWASP #3 issue. OWASP, or Open Web Application Security Project, is a project aiming to produce and gather information, articles, methodologies, documentation, and tools about online web application security. One of their most popular documents is the OWASP Top 10, which each year lists the most important vulnerabilities in online web software. The one we are addressing in this document comes third, after SQL Injection methods and Cross-Site Scripting (XSS).
OWASP's description of this issue is as follows:

"Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, session tokens, or exploit other implementation flaws to assume other users' identities."

This is in fact the case for almost any HTTP application, since HTTP is not a statefull protocol — no session state is kept between two different requests coming from the same user (who, most of the time, is using the same service). So session state (containing information like authentication status, must be dealt with user-side, by sending a so-called cookie

As per OWASP's methodology, the vulnerability is divided in five parts:

- Thread Agents: Consider anonymous external attackers, as well as users with their own accounts, who may attempt to steal accounts from others. Also consider insiders wanting to disguise their actions.
  - Our scenario: a user who wants to hijack another user's authenticated session.

- Attack vectors: Attacker uses leaks or flaws in the authentication or session management functions (e.g., exposed accounts, passwords, session IDs) to impersonate users.
  - Our scenario: we will exploit the cookies sent in cleartext containing all session information

- Security weakness: Developers frequently build custom authentication and session management schemes, but building these correctly is hard. As a result, these custom schemes frequently have flaws in areas such as logout, password management, timeouts, remember me, secret question, account update, etc. Finding such flaws can sometimes be difficult, as each implementation is unique.
  - Our scenario: web developpers build session management protocols manually using cookies sent by the user. Such schemes are usually flawed and rely on the cookie not being intercepted or modified by anybody (which is clearly not the case in open wireless networks)

- Technical impacts: Such flaws may allow some or even all accounts to be attacked. Once successful, the attacker can do anything the victim could do. Privileged accounts are frequently targeted.
  - Our scenario: if the service is vulnerable, then all accounts can be compromised in this fashion. Some special, less frequented websites can also be fine-targeted if information provided is valuable for the attacker.

- Business impacts: Consider the business value of the affected data or application functions. Also consider the business impact of public exposure of the vulnerability.
  - Our scenario: data obtained can usually lead to identity theft. The image of such large companies (Facebook, Twitter, etc) can also be degraded by the exposure of such a vulnerability. In fact, this vulnerability has already been widely spoken of, and some companies are slower to react than others...

It is a pretty important issue, and can affect everybody who uses popular websites such as Facebook, Twitter, Google, Blogger, etc... •

OWSAP's Logo

Fig. 2 of 5 Back to Album - CookieMonster's Profile Previous Next

CookieMonster

Wall Home Info **Profile** Privacy Share

Wall
Info
Photos (29)
Discussions

1,242,922
people like this

Likes

Sesame Street

Likes

Cookies

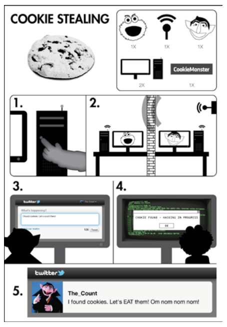
Create a Page
Report Page
Share

Profile: Our attack. Cookiemonster focuses mainly on cookie snarfing to get session information and therefore hijack ongoing sessions with popular web services such as email providers, blogs, social networks, etc...

It sniffs on an wifi interface, sniffs out interesting packets (the ones containing cookies), strips out the cookiesand spawns web views of the websites for which the cookies were detected.

As you may know, Cookie Monster is a Muppet on the tv show Sesame Street. He is known for his appetite, his love of cookies and his famous phrase: «Me want cookie! Om nom nom nom!». In 1972, during a violent altercation, he quarreled with the count, another Sesame Street character, who would sometimes prefer to count cookies (see what we did there?) instead of eating them. We will use this well-known rivalry to describe our program's features.

CookieMonster
User Guide


COOKIE STEALING
1X
1X
1X
2X
CookieMonster
1X
1.
2.
3.
4.
COOKIE FOUND - HACKING IN PROGRESS
5.
twitter
The_Count
I found cookies. Let's EAT them! Om nom nom nom!

May 2 at 2:38pm – Like – Comment – Share

CookieMonster Find an enlargement of this image on page 6.
48 minutes ago – Like

CookieMonster What you need: A victim (the Count), An attacker (the Cookiemonster), An open wifi network (public parcs, coffee shops, airports...), Two computers (victim computer and attacking / sniffing computer), One instance of the cookiemonster script running on the sniffing computer
46 minutes ago – Like

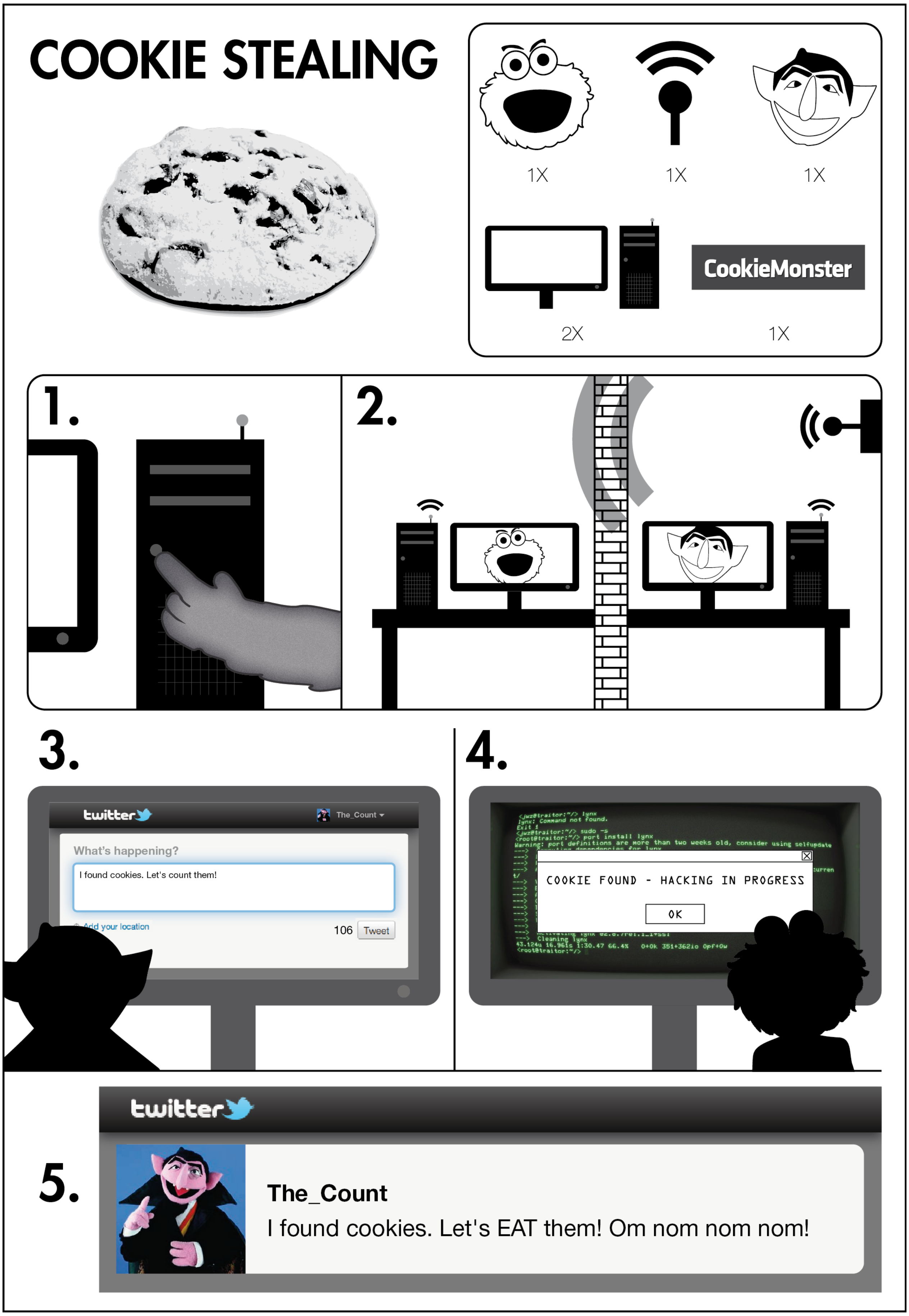
CookieMonster 1 & 2: Make sure both computers are turned on and working. Both computers must be within range of the wifi network and associated to the access point.
(NB: for the attack to work, we actually only need to be close to the victim computer – as long as our wireless interface is in monitor mode, we only need to capture the packets SENT to the access point – we don't even need to be associated to the same wireless network, although it might be necessary depending on how the session is managed server-side. More on this later.)
43 minutes ago – Like

CookieMonster 3 & 4: Let the victim browser around his or her favorite websites – email, social networks, blogs, etc. The cookies will be captured on-the-fly (depending on which version of cookiemonster is being used). Each cookie found will be reused to open the corresponding webpage using the victim's credentials.
37 minutes ago – Like

CookieMonster 5: This yeilds total control over the user's account on that website, giving us possibility to do everything the user could do. This can go from posting annoying and immature status updates to recovering credit card numbers. Possibilities are infinite, impossible is nothing. And you just did it. You're lovin' it, right? Connecting people, I mean. Life's good.
33 minutes ago – Like

User Guide

Fig. 3 of 5 Back to Album – CookieMonster's Profile Previous Next



I found cookies. Let's count them!

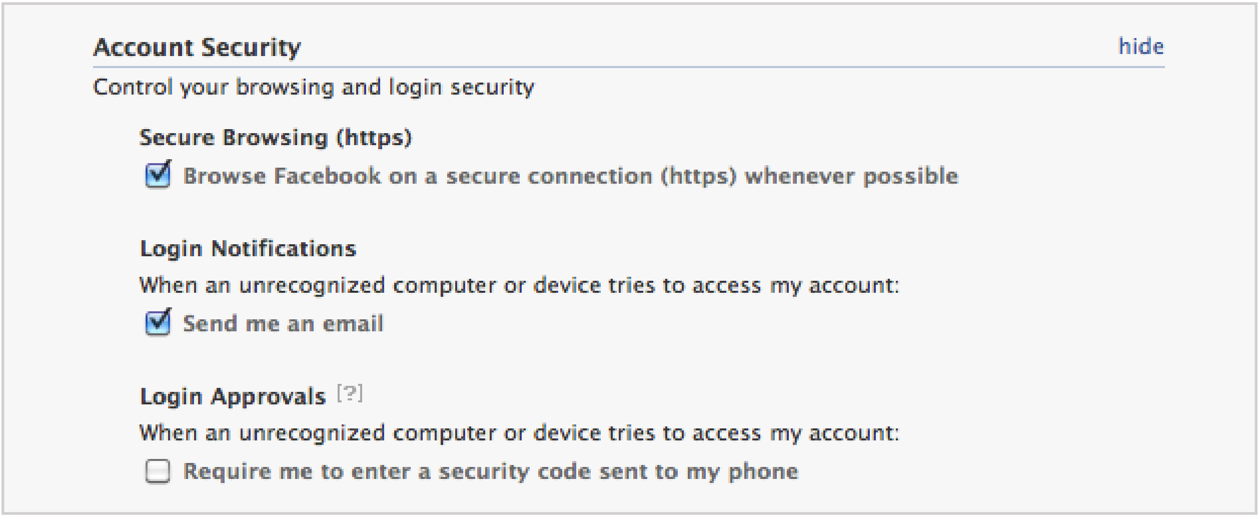
```
<jwz@traitor:~/> lynx
lynx: Command not found.
Exit 1
<jwz@traitor:~/> sudo -s
<root@traitor:~/> port install lynx
Warning: port definitions are more than two weeks old, consider using selfupdate
--->
--->
--->
--->                                                                  curren
t/
--->
--->
--->
--->
--->
--->
--->
--->
--->
--->
---> Cleaning lynx
43.124u 16.961s 1:30.47 66.4%   0+0k 351+362io 0pf+0w
<root@traitor:~/>
```

COOKIE FOUND - HACKING IN PROGRESS

The_Count
I found cookies. Let's EAT them! Om nom nom nom!

Facebook's account security settings

Fig. 4 of 5 Back to Album – CookieMonster's Profile Previous Next


Account Security hide

Control your browsing and login security

Secure Browsing (https)

✓ Browse Facebook on a secure connection (https) whenever possible

Login Notifications

When an unrecognized computer or device tries to access my account:

✓ Send me an email

Login Approvals [?]

When an unrecognized computer or device tries to access my account:

☐ Require me to enter a security code sent to my phone

Privacy: How to protect against it. So, all of this is pretty scary right? Fortunately, there are many ways that can be used to avoid this. Some work better than others, some are simpler to implement and use. Some are client-side, other can be used server-side.

Client-side.
The main problem that makes cookie snarfing possible is the fact that the cookie is sent by the client to the server in cleartext. The solution — you might have an idea — is to encrypt the data transiting to the server. This can be accomplished in a variety of ways:

• Using proper wireless encryption: this will make your data unreadable by any eventual eavesdropper. Be reminded that WEP is not sufficiently secure for this - anyone in possession of the WEP key can easily break encryption, and anyone not possessing the WEP key can recover it in under 10 minutes. WPA and WPA2 are good choices, since traffic to and from the access point is encrypted per-device. That being said, if the attacker knows the password (pre-shared key) to the WPA or WPA2 network, then it is possible to decrypt traffic if all four packets from the EAPOL exchange between the target device and access point have been captured.

• Using SSL (HTTPS) when available: this will activate point-to-point encryption between your computer and the remote server. Really hard to use (impossible, actually) if the remote server does not have an HTTPS version. This is [sic] the case for most websites, even though a few minority (like Facebook and Google) do offer the possibility to experience their whole site in a secure fashion.

• SSH tunnels, VPNs, and encrypted proxies: these are the easiest options to implement for the tech-savvy. The only thing you need is an external server running an SSH server (this could even be Chalmers or your home university). Use an SSH client to set up a proxy on your local computer and all data between your computer and the SSH server will be encrypted. VPNs offer the same level of protection, but can be harder to configure server-side. You could also use encrypted proxys such as Tor (The Onion Router) which encrypt and anonymize connections. All these services offer the same level of protection, and should work in a wireless LAN, as long as the necessary ports are open.

Server-side.
• Secure Sockets Layer (SSL): Security conscious web developers should always implement an HTTPS version of their websites for users who are concerned with their privacy.

• Smart cookie-management: Web developers should not include any more information than necessary in the cookie (cleartext passwords or encrypted passwords, user IDs, etc...), always assuming it is being send in cleartext and can be easily intercepted. Furthermore, session management should be carfully thought out, not only relying on the session ID contained in the cookie for identification and authentication. More information could be taken into account, like IP addresses or user-agents. These can ultimately be forged, but it is still one further step to perform for the attacker. Extra cookies should be transmitted and set only during HTTPS sessions, so that even if the attacker captures some cleartext cookies, he will need the HTTPS cookies to successfully hijack the session.

Share: How to extend it. Cookiemonster's abilities are quite limited at the time of this writing. It can only sniff cookies from open wireless networks. Further improvements could include ARP poisoning, in order to circumvent wifi encryption (why go through the trouble of decrypting traffic when the access point can do it for us?). Of course, this would require to associate and authenticate to the access point, but it still remains a possibility if WPA is used in public places and the key not changed frequently enough.

ARP poisoning deals effectively with link layer protection, but not with application-level encryption. If ARP poisoning goes both ways, fake certificates could be generated and dealt to both parties, effectively placing cookiemonster in a monster-in-the-middle (MITM) scenario. This will issue warnings to the victim(s), which will be completely ignored in most cases.

Wireless (in)security is not a very hot topic, when it actually should be. The security implications of broadcasting traffic allows for a whole new set of attack vectors that wouldn't have been possible with the uses of good-ol' Ethernet cables and switched LANs. cookiemonster is just a needle in a haystack, and it can sting a lot when used in public places. We are really looking forward to extend its capabilities regarding eavesdropping on "lambda user" internet usage (web pages, emails, etc). This would hopefully help the IT industry to provide secure communications out-of-the-box for everybody. •

Running CookieMonster

Fig. 5 of 5 Back to Album – CookieMonster's Profile Previous Next

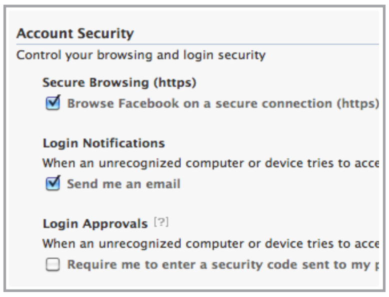
The cookiemonster script has two main MOs. One that operates by sniffing directly on a network interface (useful when the wireless adaptor can be put in monitor mode) and another one that "sniffs" a pcap file (useful when capturing from another computer using more flexible network sniffers, like Wireshark).

Cookiemonster was conceived and tested with the latest version of scapy, an extremely powerful low-level network API for python, and python 2.6.5. These are available in the security-oriented Linux distribution Backtrack 5.

As we said, this is a work-in-progress. This program is provided as is and might not work in certain circumstances, or may have a bug or two. As usual, it has been created for educational purposes only, and we do not take any responsibility whatsoever for any misuse of this tool or the consequences entailed by such actions or other actions performed with it. (So no messin' around).

Figures index

By CookieMonster – Updated about 1 hour ago


Account Security
Control your browsing and login security

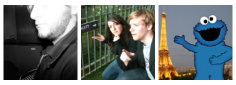
Secure Browsing (https)
☑ Browse Facebook on a secure connection (https)

Login Notifications
When an unrecognized computer or device tries to acce
☑ Send me an email

Login Approvals [?]
When an unrecognized computer or device tries to acce
☐ Require me to enter a security code sent to my

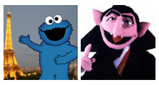
Add Profile Picture | Tag Photos

In This Album (3) See All



Event Invitations See All

Cookie Party
Friday, May 27



RSVP: Yes – No – Maybe

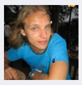
Sponsored See All

It could happen to you!

Identity theft is not a joke. Millions of families suffer every year. Make sure that you are protected enough against it!

Like · 1,153 people like this.

Like – Share


Write a comment...

Share this album with anyone by sending them its public link