WEB APPLICATION SECURITY 101

JACK SCOTT
@JACKSCOTTAU

WHY SHOULD YOU LISTEN TO ME?

- I run Loop Foundry, a software and web development consultancy.
- I've been writing web-based software professionally for ten years now.
- I read lots about the Internet and web security.
- I care about you.

- But most of all…

I HAVE MADE EVERY SINGLE ONE OF THESE MISTAKES.

WHY SHOULD YOU CARE?

THE INTERNET HATES YOU

WELL, NOT YOU SPECIFICALLY…

IT HATES EVERYBODY!

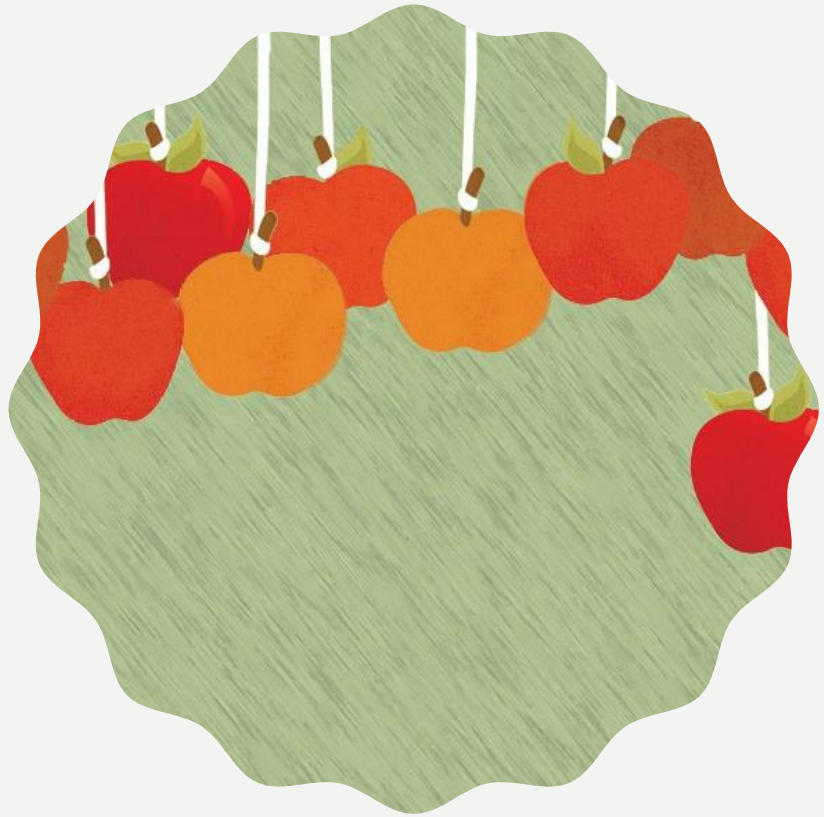
- Hackers will try to attack whatever they can.
- They'll concentrate on the things with the lowest hanging fruit:
  - Easiest to hack.
  - Juiciest results.

- **Because web security risks are business risks and business risks are legal and financial risks.**

WHAT DO WE HAVE TO DO?

- If you have low hanging fruit, somebody will pick them.
- If all your security fruit are located high up on the tree, potential attackers might try another tree.

- It's about reducing your target area for potential attacks.
  - Bigger target = bigger risk.
  - Smaller target = smaller risk.



WHAT ARE WE TRYING TO PROTECT?

- Your user's data:
  - Credit card details
  - Passwords and other authentication details
  - Personal identification data
  - And more!
- Your application's code
- Your company's data

TRANSPORT SECURITY

WHY ENCRYPT USING SSL/TLS?

- Prevents man-in-the-middle attacks from people eavesdropping the connection:
  - ISPs and Governments
  - Employers of your users (to some degree)
- Ensures that the content of the page cannot be changed while in transit.
- Your website being SSL-encrypted has SEO benefits too, if that's your thing.

- Use *everywhere* on your site, not just on login or payment pages.

HOW TO SSL…

- **Option 1 - LetsEncrypt**
- If you're hosting your application or site on a standard Linux box using a web server such as Apache or Nginx: Just install LetsEncrypt. It's free, automated, and extremely easy to set up.
- **Option 2 – Cloud-provided Certificates**
- If you're hosting something using Amazon AWS or Azure (especially if using a content delivery network), you can get one from them for either free or very cheap.
- **Option 3 – Paid Certificates**
- If your needs are more complex, or you use IIS on Windows, you can still pay money for a certificate from companies such as Symantec or Verisign.

SSL Report: loopfoundry.com.au (35.165.54.24)

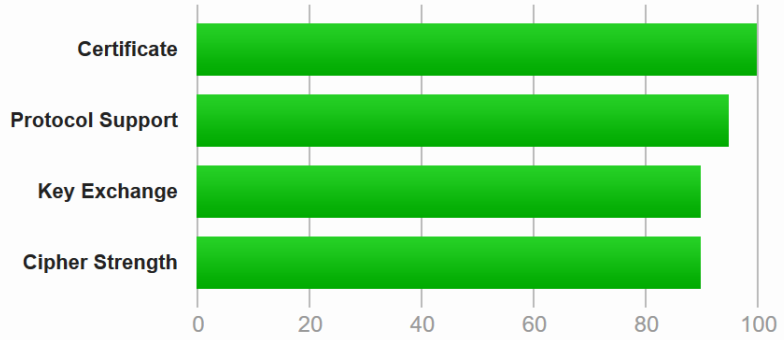
Assessed on: Wed, 30 Aug 2017 00:18:36 UTC | Hide | Clear cache

Scan Another »

Summary

Overall Rating

A



Visit our documentation page for more information, configuration guides, and books. Known issues are documented here.

Certificate #1: RSA 2048 bits (SHA256withRSA)

Server Key and Certificate #1

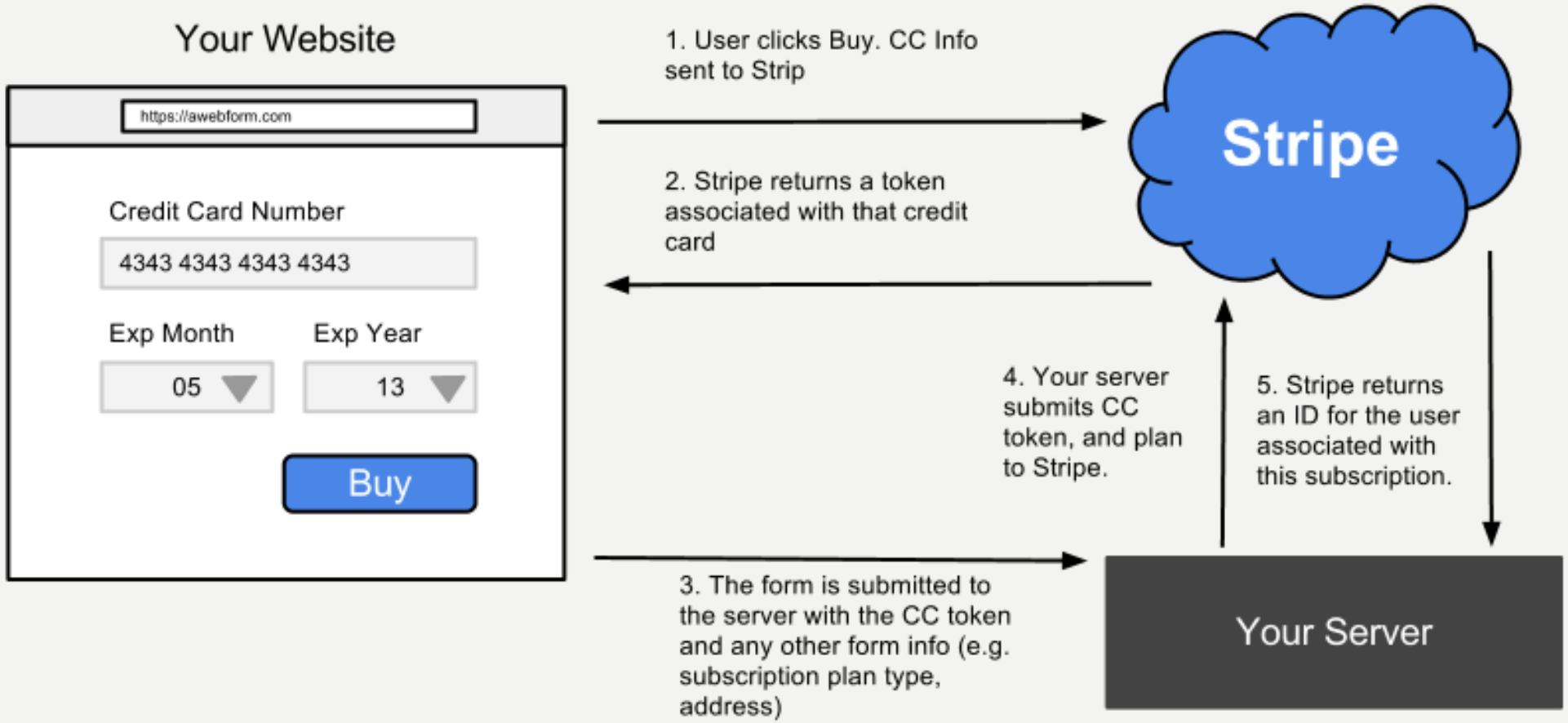
CREDIT CARDS

WHY NOT STORE CREDIT CARD DATA?

- PCI-DSS Compliance (www.pcicomplianceguide.org)
  - Applies to all merchants, online and offline, who deal with credit card payments.
- Multiple 'levels':
  - Level A for merchants who outsource payments (such as Stripe).
  - Level C for merchants who deal with in-person payments.
  - Level D for merchants who process credit cards themselves.

- The golden rule: **You do not want a credit card number to come anywhere near your server. Ever.**

HOW TO HANDLE CREDIT CARD PAYMENTS…



http://liamkaufman.com/blog/2012/11/09/stripe-would-be-perfect-if/

PASSWORDS

DON'T DO THESE THINGS...

1

Don't send them plain text in an email.

- No, I don't care what your UX guy or your client says.

2

Don't store them plain text.

- They will end up in a database dump on somebody's laptop at some point.

3

Don't restrict to a maximum length or particular character set for no good reason.

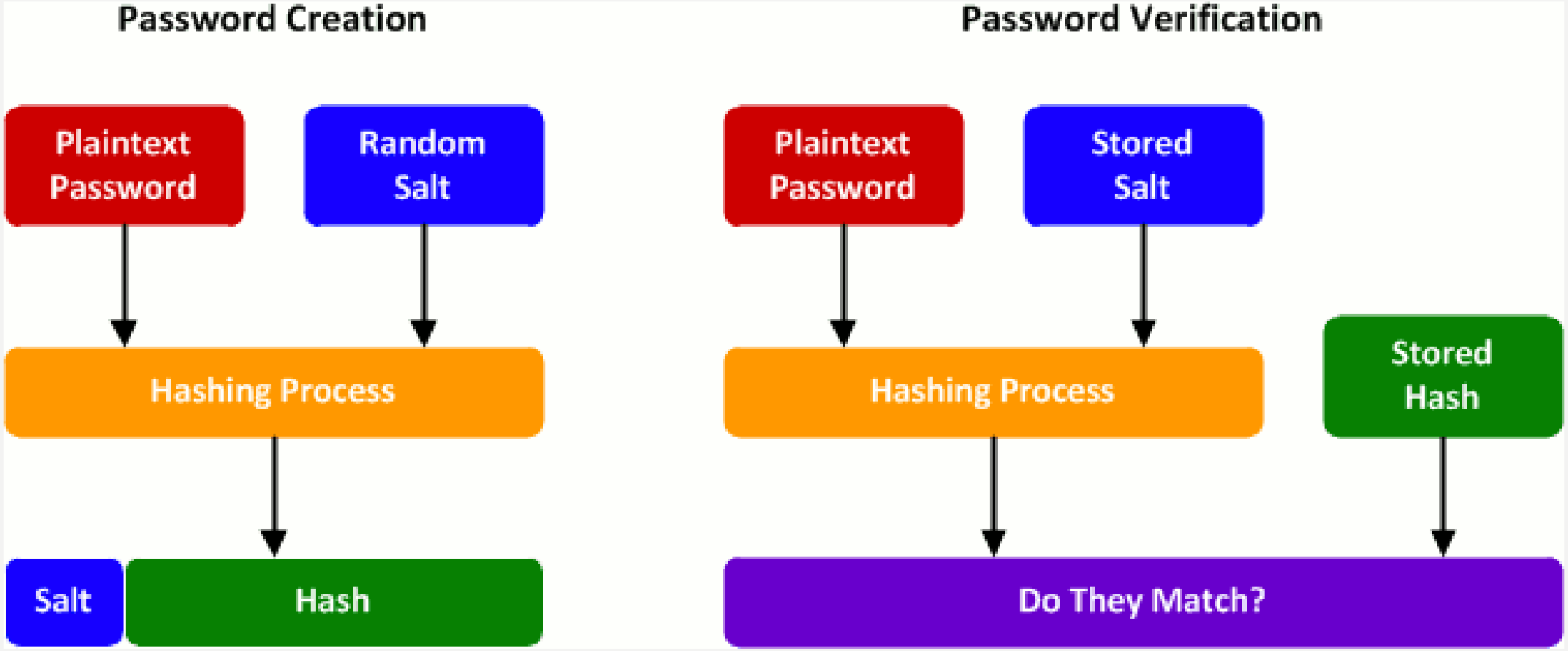
4

Don't roll your own cryptography!

DO THESE THINGS…

Store	Always store passwords in a form where they cannot be retrieved as plain text (hashed).
Restrict	Specify a minimum length (8 characters at least these days). •But no other restrictions!
Send	Send confirmation emails when a password is changed or reset (but don't send the password)!
Check	Make sure they aren't using "password" as their password.

PASSWORD HASHING



http://blog.conviso.com.br/worst-and-best-practices-for-secure-password-storage/

SPECIFICS

- Have three database fields for password storage:
  - Hash of password + salt
  - Raw salt
  - Algorithm used
- The best hashing algorithms are constantly changing as the hardware used to break them changes.
  - Desktops with GPUs are getting good at cracking passwords.
  - You can rent a supercomputer from Amazon for dollars an hour these days.
- If your framework offers something, it's *probably* your best bet – but do due diligence.

ALTERNATIVES TO PASSWORDS

- Security questions?
  - My mother's maiden name is searchable on the Internet.
  - So is my first job and where I grew up.
- OAuth Login
  - Using Facebook / Google / Twitter accounts as authentication mechanisms
  - Appropriate in some cases, not in others.
- Two-Factor Authentication
  - U2F (like a smart card)
  - Time-based One Time Password (Google Authenticator)

WHAT ABOUT USER-ENTERED DATA?

WHAT DO YOU ACTUALLY NEED TO KNOW?

- Do you need to know their legal first and last names?
  - Or just what they want to be called?
- Do you need their gender (or sex)?
- Do you need their address?
- Do you need their birthday?

- This is incredibly valuable information to the user, whether they know it or not.

STAFF ACCESS

- Your staff might need to masquerade as a user for diagnostic purposes.
- You might want to collate user data for reporting purposes.

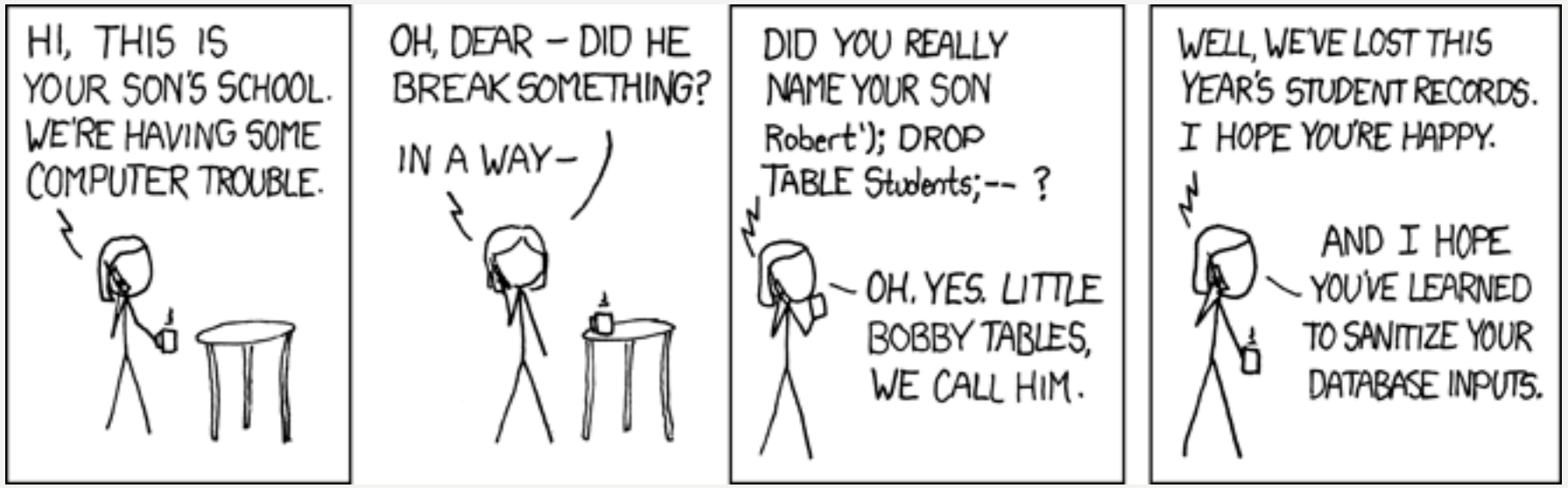
- Can you log or otherwise audit which of your staff have accessed what customer data, and at what time?
  - If not, why not?
  - Is it appropriate to let users know when their data has been accessed?

THINK LIKE AN ATTACKER

BACKUPS AND LOG FILES

- Where are your backups kept?
  - Are they encrypted?
  - Which of your staff have access to them?
  - Are they publicly accessible?
  - While I'm here: Have you actually tested them?
- What about log files?

SQL INJECTION



https://xkcd.com/327

API ENDPOINTS

- Do you actually check to make sure that a user has authorization to access a particular endpoint?
- It sounds stupid, and it is, but make sure your tests actually test for this.

GET /API/CAT/<CATID>

CatID	UserID	CatName
1	1	Kitty
2	1	Meow Meow Fuzzy Face
3	2	Mittens

If user #2 accesses cat #3:
Name: Mittens

If user #1 accesses cat #3 (expected result):
Forbidden 403

But what if it did this when user #1 accesses cat #3:
Name: Mittens

SOFTWARE UPDATES

- Operating system updates
- Database management system updates
- Language and runtime updates
- Library updates

- Using an old version isn't wrong, but you should have a think about the risk vs. reward.

- A lot of this can be automated or abstracted away (such as using PaaS).

WHEN THE WORST HAPPENS

AND IT PROBABLY WILL…

OWN UP TO IT

- You won't know until afterwards.
  - Almost all attacks are discovered later on, and usually because the attacker is selling data.
  - This can be *years* after the original attack.
- Apologise to your users and be honest with them.
- Fix whatever the problem was, and put in place tests to make sure it doesn't happen again.

FINAL THOUGHTS

PAYING SOMEBODY

- Can you hire a white-hat hacker to try and get access to your system?
- Can you hire a security consultant to advise you on what is right for you?
- From a business point of view, it's like insurance – it's only got to pay once.

TL;DR:

01

Don't store any data you don't have to.

02

When you do have to store something, take care of where you put it.

03

When you have to move something, make sure it's encrypted.

04

Have tests to **ensure** that things are as they should be.

05

When you screw up, own up to it.

NOW, HOW ARE WE GOING FOR TIME?

JACK SCOTT (@JACKSCOTTAU)