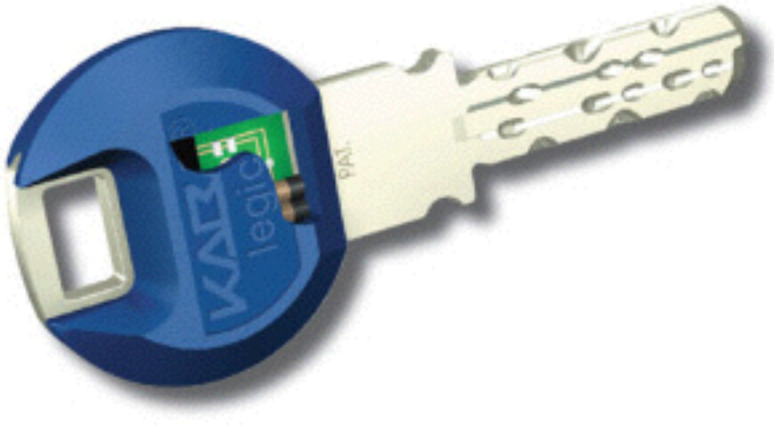
Securing your Spring App with TOTP



Pablo Caif

shine technologies

Agenda

- The problem we are trying to solve
- MFA - Multifactor authentication
- TOTP - Time based One time Password
- Spring Security
- Token Generators Apps
- Demo!!!

Pa55words P@ssword$ PassW0rds

- Vulnerable to Social Engineering
- They need to be long
- They need to be complex
- They could be stolen
- So many to remember

Is there any way to stop this madness?

Should we keep making our life difficult?

Agenda

- ~~The problem we are trying to solve~~
- MFA
- TOTP
- Spring Security
- Token Generators Apps
- Demo!!!

MFA

- Something that you know
- Something that you have
- One time password (OTP)

OTP

- RSA Tokens
- SMS Tokens

Enter SMS code

After a while

Enter SMS code

SMS Message
235587

Wrong code
Enter SMS code

SMS Message
784593

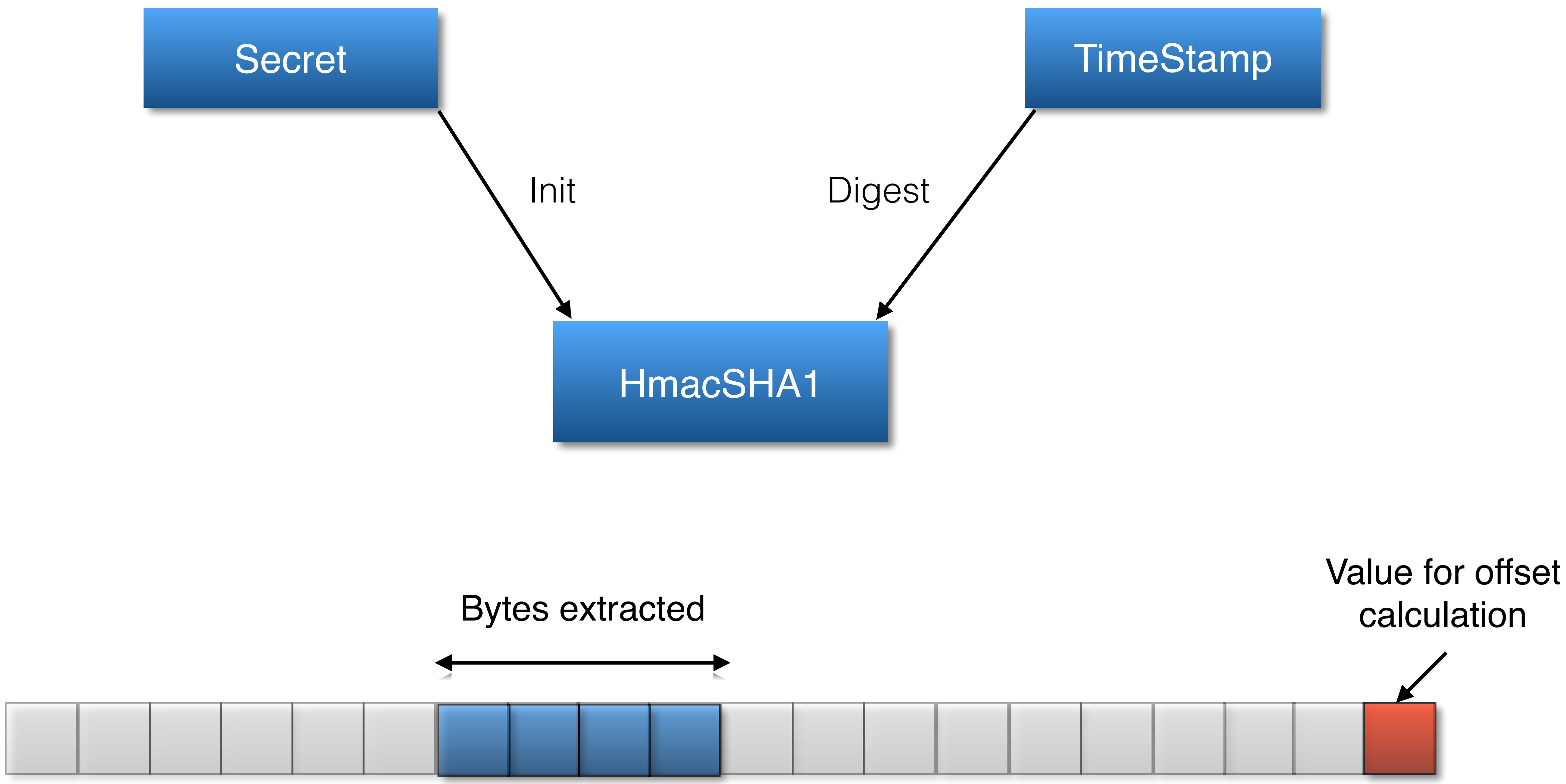
Agenda

- ~~The problem we are trying to solve~~
- ~~MFA~~
- TOTP
- Spring Security
- Token Generators Apps
- Demo!!!

TOTP

- Standard open source algorithm
- Shared secret key
- Time based
- Implemented by e.g. Google authenticator, Authy…
- You can implemented yourself

How does it work?



The algorithm

```
public long getCode(byte[] secret, long timeIndex)
            throws NoSuchAlgorithmException, InvalidKeyException {
    SecretKeySpec signKey = new SecretKeySpec(secret, "HmacSHA1");
    //We put the timeIndex in a bytes array
    ByteBuffer buffer = ByteBuffer.allocate(8);
    buffer.putLong(timeIndex);
    byte[] timeBytes = buffer.array();

    //Calculate the SHA1
    Mac mac = Mac.getInstance("HmacSHA1");
    mac.init(signKey);
    byte[] hash = mac.doFinal(timeBytes);

    //Calculate the offset we will use to extract our pin
    int offset = hash[19] & 0xf;
    //Clear the signed bits
    long truncatedHash = hash[offset] & 0x7f;
    //Use bits shift operations to copy the remaining 3 bytes from the
array
    //and construct our number
    for (int i = 1; i < 4; i++) {
      truncatedHash <<= 8;
      truncatedHash |= hash[offset + i] & 0xff;
    }
    //Truncate to 6 digits
    return truncatedHash % 1000000;
  }
}
```

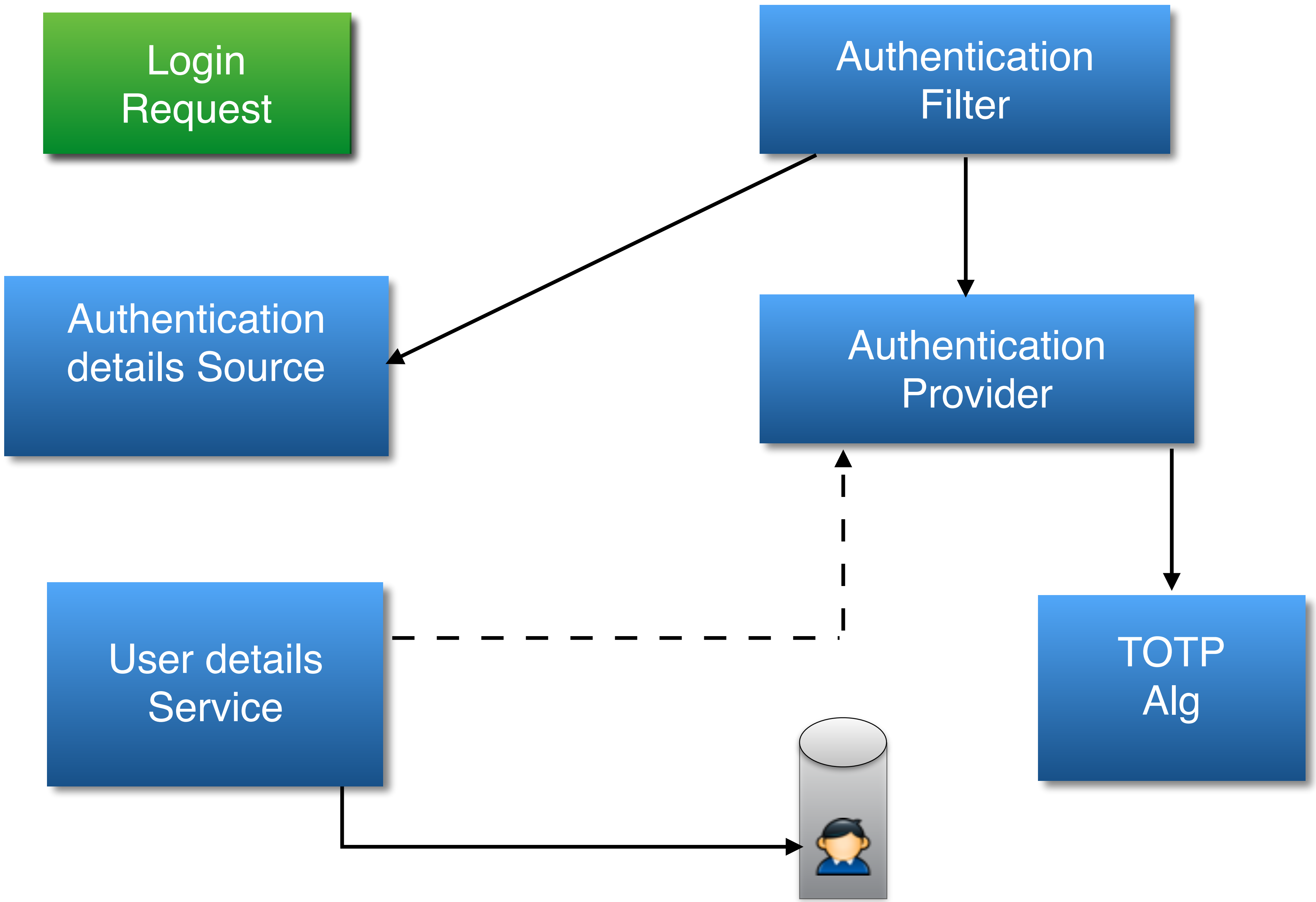
Agenda

- ~~The problem we are trying to solve~~
- ~~MFA~~
- ~~TOTP~~
- Spring Security
- Token Generators Apps
- Demo!!!

Spring Security

- Spring module
- Deals with the complexity of authentication and authorisation
- Fully customisable
- Need to know it in depth to customise it

Spring security authentication process



Authentication details source

```
public class TOTPWebAuthenticationDetails extends WebAuthenticationDetails {
  private static final long serialVersionUID =
SpringSecurityCoreVersion.SERIAL_VERSION_UID;
  private Integer totpKey;

  public TOTPWebAuthenticationDetails(HttpServletRequest request) {
      super(request);
      String totpKeyString = request.getParameter("TOTPKey");
      if (StringUtils.hasText(totpKeyString)) {
        try {
          this.totpKey = Integer.valueOf(totpKeyString);
        } catch (NumberFormatException e) {
          this.totpKey = null;
        }
      }
  }

  public Integer getTotpKey() {
    return this.totpKey;
  }
}
```

User details service

```
@Component
public class DBUserDetailsService implements UserDetailsService {

  @Autowired
  private UserRepository userRepository;

  @Override
  public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
    DBUser user = userRepository.findOne(username);
    if (user == null) {
      throw new UsernameNotFoundException(username);
    }
    return new TOTPUserDetails(user);
  }
}
```

```
public class TOTPUserDetails implements UserDetails {
  private String username;
  private String password;
  private boolean enabled;
  private String secret;
  private Collection authorities = new HashSet<>();

  ...
  ...
  public TOTPUserDetails(DBUser user) {
    this.username = user.getUsername();
    this.password = user.getPassword();
    this.enabled = user.isEnabled();
    this.secret = user.getSecret();
    populateAuthorities(user.getRoles());
  }
  …
}
```

Authentication provider

```
public class TOTPAuthenticationProvider extends DaoAuthenticationProvider {
  private TOTPAuthenticator totpAuthenticator;

  @Override
  protected void additionalAuthenticationChecks(UserDetails userDetails,
                                                  UsernamePasswordAuthenticationToken authentication)
            throws AuthenticationException {

    super.additionalAuthenticationChecks(userDetails, authentication);

    if (authentication.getDetails() instanceof TOTPWebAuthenticationDetails) {
      String secret = ((TOTPUserDetails) userDetails).getSecret();

      if (StringUtils.hasText(secret)) {
        Integer totpKey = ((TOTPWebAuthenticationDetails) authentication
                        .getDetails()).getTotpKey();
          if (totpKey != null) {
            try {
              if (!totpAuthenticator.verifyCode(secret, totpKey, 2)) {
                throw new BadCredentialsException("Invalid TOTP code");
              }
            } catch (InvalidKeyException | NoSuchAlgorithmException e) {
                throw new InternalAuthenticationServiceException("TOTP code verification failed", e);
            }
          } else {
              throw new MissingTOTPKeyAuthenticatorException("TOTP code is mandatory");
          }
      }
    }
  }
 }
```

Agenda

- ~~The problem we are trying to solve~~
- ~~MFA~~
- ~~TOTP~~
- ~~Spring Security~~
- Token Generators Apps
- Demo!!!

Token generator apps

- They just implement the same algorithm
- They know your secret key and the time
- They are the ‘Something that you have’
- Mobile apps are the best example
- You can implement your own

Demo time !!!

pablocaif@gmail.com
Github: https://github.com/pablocaif/TOTP-spring-example

Thank you

Questions?