PEMANFAATAN TELEGRAM BOT API DALAM LAYANAN OTENTIKASI TANPA PASSWORD MENGGUNAKAN ALGORITMA *TIME-BASED ONE-TIME PASSWORD* (TOTP)

ARTIKEL SKRIPSI

Diajukan Untuk Memenuhi Sebagian Syarat Guna
Memperoleh Gelar Sarjana Komputer (S.Kom.)
Pada Program Studi Teknik Informatika

OLEH:

UKAZA PUTRO SATYA PERDANA

NPM: 12.1.03.02.0191

FAKULTAS TEKNIK (FT)

UNIVERSITAS NUSANTARA PERSATUAN GURU REPUBLIK INDONESIA

UN PGRI KEDIRI

2016

Skripsi oleh:

Ukaza Putro Satya Perdana
NPM: 12.1.03.02.0191

Judul:

PEMANFAATAN TELEGRAM BOT API DALAM LAYANAN OTENTIKASI TANPA PASSWORD MENGGUNAKAN ALGORITMA *TIME-BASED ONE-TIME PASSWORD* (TOTP)

Telah disetujui untuk diajukan kepada
Panitia Ujian/Sidang Skripsi Program Studi Teknik Informatika
Fakultas Teknik UN PGRI Kediri

Tanggal: 26 JULI 2016

Pembimbing I	Pembimbing II
Fatkur Rhohman, M.Pd.	Resty Wulanningrum, M.Kom.
NIDN 0728088503	NIDN 0719068702

ii

Skripsi oleh:

Ukaza Putro Satya Perdana
NPM: 12.1.03.02.0191

Judul:

PEMANFAATAN TELEGRAM BOT API DALAM LAYANAN OTENTIKASI TANPA PASSWORD MENGGUNAKAN ALGORITMA *TIME-BASED ONE-TIME PASSWORD* (TOTP)

Telah dipertahankan di depan Panitia Ujian/Sidang Skripsi
Program Studi Teknik Informatika FT UN PGRI Kediri
Pada tanggal: 11 Agustus 2016

Dan Dinyatakan Telah Memenuhi Persyaratan

Panitia Penguji:

1. Ketua Penguji : Fatkur Rhohman, M.Pd. _____
2. Penguji I : Dr. Suryo Widodo, M.Pd. _____
3. Penguji II : Resty Wulanningrum, M.Kom. _____

Mengetahui,
Dekan FT

Dr. Suryo Widodo, M.Pd.
NIP. 19640202 199103 1 002

iii

PEMANFAATAN TELEGRAM BOT API DALAM LAYANAN OTENTIKASI TANPA PASSWORD MENGGUNAKAN ALGORITMA *TIME-BASED ONE-TIME PASSWORD* (TOTP)

Ukaza Putro Satya Perdana
12.1.03.02.0191
Fakultas Teknik – Teknik Informatika
ukazap@live.com
Fatkur Rhohman, M.Pd. dan Resty Wulanningrum, M.Kom.
UNIVERSITAS NUSANTARA PGRI KEDIRI

ABSTRAK

Ukaza Putro Satya Perdana: Pemanfaatan Telegram Bot API dalam Layanan Otentikasi Tanpa Password Menggunakan Algoritma Time-Based One-Time Password (TOTP)

Penelitian ini dilatarbelakangi hasil pengamatan dan pengalaman peneliti, bahwa otentikasi berbasis password mempunyai beberapa kelemahan dalam hal keamanan dan kebergunaan (*usability*). Permasalahan penelitian in adalah: (1) Bagaimana merancang layanan otentikasi tanpa password berbasis algoritma TOTP memanfaatkan Telegram Bot API?, dan (2) Bagaimana membangun aplikasi layanan otentikasi tanpa password berbasis algoritma TOTP memanfaatkan Telegram Bot API?

Teori-teori pokok yang melandasi pengembangan ini antara lain adalah konsep otentikasi, multi-factor authentication, federated authentication, dan hubungan antara kebergunaan (*usability*) dengan keamanan.

Penelitian ini menggunakan pendekatan *Research and Development* (R&D) dengan subjek penelitian adalah mahasiswa pengguna *smartphone* yang terkoneksi dengan Internet. Penelitian dilaksanakan menggunakan secara iteratif, dimulai dengan studi kondisi otentikasi saat ini, kemudian pengembangan dengan *paper prototyping*, implementasi produk, dan evaluasi menggunakan instrumen *System Usability Scale.*

Kesimpulan hasil penelitian ini adalah: (1) Telah dihasilkan rancangan layanan otentikasi tanpa password berbasis algoritma TOTP memanfaatkan Telegram Bot API, dan (2) Telah dikembangkan aplikasi layanan otentikasi tanpa password berbasis algoritma TOTP memanfaatkan Telegram Bot API.

Kata Kunci: Otentikasi, Usability, Time-based One-time Password, Telegram Messenger

I. LATAR BELAKANG

Sistem login menggunakan *password* sudah usang sejak lama, ibarat telepon umum yang bisa ditemukan di dekat restoran, *password* sudah tidak diperlukan bertahun-tahun, sebab terdapat dua teknologi mainstream yang telah menjadikan *password* usang, yakni SMS dan *e-mail*, kemampuannya mengirim e-mail atau SMS kepada user dengan cepat dan handal (Balthrop, 2014) yang dapat dimanfaatkan sebagai faktor tambahan (faktor *possession/ownership*) dalam *multi-factor authentication*, di mana setelah memasukkan *username* dan *password*, pengguna diminta memasukkan kode yang dikirimkan ke telepon seluler atau inbox e-mailnya. Tentu saja ini membantu mengatasi masalah keamanan yang dimiliki *password*, namun masalah *usability*/kebergunaan masih belum teratasi, bahkan semakin buruk dengan bertambahnya langkah untuk dapat masuk ke dalam suatu sistem. Aspek kebergunaan dalam otentikasi inilah yang menjadi perhatian utama dalam penelitian ini, di mana faktor otentikasi berbasis pengetahuan/*knowledge* perlu diminimalisasi atau bahkan ditiadakan. Dalam penelitian ini diharapkan sistem login baru berbasis *time-based one-time password* (TOTP) yang dibangun dengan memanfaatkan *Telegram* memiliki keunggulan dalam hal kebergunaan dibandingkan dengan sistem yang sudah ada.

Berdasarkan latar belakang masalah yang dipaparkan atas, maka dapat diambil rumusan masalah dari penelitian ini yaitu: (1) Bagaimana merancang layanan otentikasi tanpa password berbasis algoritma TOTP memanfaatkan Telegram Bot API? dan (2) Bagaimana membangun aplikasi layanan otentikasi tanpa password berbasis algoritma TOTP memanfaatkan Telegram Bot API?. Sehingga tujuan dari penelitian ini adalah (1) Menghasilkan rancangan layanan otentikasi tanpa password berbasis algoritma TOTP yang memanfaatkan Telegram Bot API; dan (2) Membangun aplikasi layanan otentikasi tanpa password berbasis algoritma TOTP yang memanfaatkan Telegram Bot API.

II. METODE

Algoritma *Time-Based One-Time Password* (TOTP) merupakan pengembangan dari algoritma *HMAC-based One-Time Password* (HOTP), yang dibuat untuk mendukung *moving factor* (faktor bergerak) berbasis waktu. HOTP menetapkan algoritma OTP berbasis *event*, dimana faktor bergeraknya adalah *event counter*. Varian berbasis waktu (TOTP) menyediakan nilai OTP dengan masa berlaku singkat, yang menguntungkan untuk meningkatkan keamanan. TOTP dapat digunakan pada banyak jenis aplikasi jaringan, dari akses *Virtual Private Network (VPN)* dan *login WiFi*, hingga aplikasi web berorientasi transaksi (M'Raihi dkk., 2011).

Dalam industri, algoritma ini sudah dimanfaatkan pada aplikasi *Google Authenticator* untuk otentikasi dua faktor (Rosetta Code, 2015). Algoritma ini dapat diimplementasikan dalam pengiriman notifikasi dan verifikasi terhadap otentikasi pengguna pada sistem yang akan dibangun. Di bawah ini dijelaskan prinsip kerja algoritma TOTP:

A. Prasyarat Algoritma

1. Klien dan verifikator (server otentikasi/validasi) harus mengetahui atau bisa memperoleh *Unix time* terkini (yakni jumlah detik yang berlalu sejak tengah malam 1 Januari 1970 dalam zona waktu UTC) untuk pembangkitan OTP. Ketepatan waktu pada sisi klien mempengaruhi seberapa sering sinkronisasi jam harus dilakukan.
2. Baik klien maupun verifikator harus berbagi rahasia yang sama atau pengetahuan tentang bentuk transformasi untuk membangkitkan rahasia bersama.
3. Algoritma harus menggunakan HOTP sebagai komponen pembangun.
4. Klien dan verifikator harus menggunakan nilai *time-interval* yang sama.
5. Harus ada rahasia unik (kunci) untuk setiap klien yang berbeda.
6. Kunci harus dibangkitkan secara acak atau diturunkan menggunakan suatu algoritma penurunan kunci.
7. Kunci boleh disimpan dalam perangkat aman dan harus dilindungi dari akses dan penggunaan yang tidak sah.

B. Tahap Inisiasi

Ada dua tahap dalam algoritma TOTP, yang pertama bisa disebut sebagai tahap inisiasi dan yang kedua adalah tahap penghitungan TOTP (dilakukan saat melakukan otentikasi).

Berdasarkan *RFC 6238* (M'Raihi dkk., 2011) Implementasi acuan untuk inisiasi adalah sebagai berikut (Rosetta Code, 2015):

1. Bangkitkan kunci K, yakni *bytestring* sembarang, dan bagikan secara aman dengan klien.
2. Sepakati *epoch* T0 dan sebuah interval TI, yang akan digunakan untuk menghitung nilai dari *counter* C (umumnya adalah T0 = *Unix epoch* dan TI = 30 detik).
3. Sepakati fungsi *hash* kriptografi yang digunakan (umumnya *SHA-1*).
4. Sepakati panjang token N (umumnya 6 digit).

C. Tahap Penghitungan OTP

Setelah atribut-atribut di atas diketahui dan disepakati pada tahap inisiasi oleh kedua pihak yakni klien dan verifikator, selanjutnya penghitungan *one-time password* dilakukan dengan langkah-langkah berikut (Krall, 2013):

a. Hitung *message authentication code* menggunakan fungsi *hash-based message authentication code (HMAC)* sesuai metode *hash* yang disepakati (misal *SHA-1*) dengan parameter kunci rahasia *K* dan waktu saat ini *T0* dibagi interval *T1*, fungsi HMAC-SHA1 didefinisikan sebagai (Houtven, 2014):

$$HMAC(K,m) = SHA1((K \oplus P_{inner}) \,||\, SHA1(K \oplus P_{outer}) \,||m)$$

Di mana:

1) *SHA1* adalah algoritma/fungsi hash kriptografi SHA-1,
2) *K* adalah kunci rahasia,
3) *m* adalah pesan yang akan di-otentikasi (T0 / T1),
4) || adalah operasi *concatenation* atau penyambungan,
5) $\oplus$ adalah operasi *exclusive (XOR),*
6) P_{inner} adalah konstanta *inner padding* bernilai 0x3636..36 (sepanjang ukuran blok *digest* SHA1: 160 bit),
7) P_{outer} adalah konstanta *outer padding* bernilai 0x5c5c..5c (sepanjang ukuran blok *digest* SHA1: 160 bit).

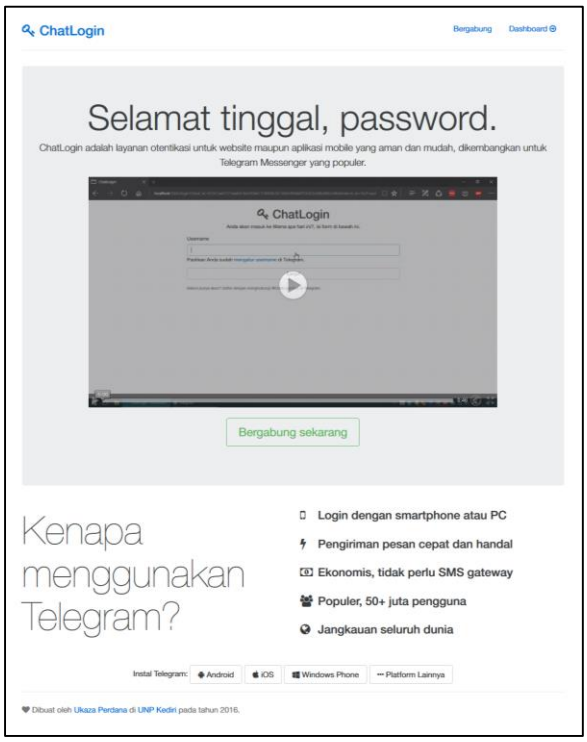
Misal diperoleh $HMAC\left(K, \frac{T0}{T1}\right) = 1f8698690e02ca16618550ef7f19da8e945b555a$

b. *Message authentication code* berukuran 160-bit di atas sudah bisa digunakan namun bisa disederhanakan lagi menggunakan *dynamic truncation* supaya mudah dibaca oleh manusia:

1) Tentukan *offset* yakni nilai 4 bit bawah dari *byte* terakhir. *Byte* terakhir adalah $5a$, sehingga 4 bit bagian bawah bernilai a (heksademal) = 10 (desimal).
2) Ambil 4 byte dimulai dari *offset* 10: Didapat heksadesimal $50ef7f19$.
3) Konversikan menjadi bilangan desimal: 1357872921
4) Tampilkan *N* digit terakhir (sesuai yang disepakati = 6 digit): 872921. Inilah kode yang akan dikirim oleh pengguna dan diverifikasi oleh server otentikasi.

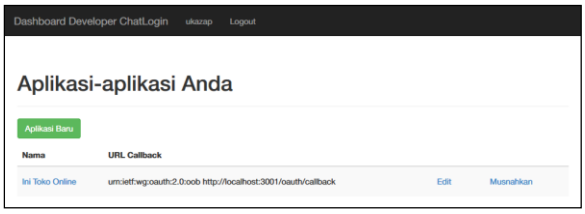
III. HASIL DAN KESIMPULAN

Berikut adalah desain akhir dari produk yang dikembangkan:

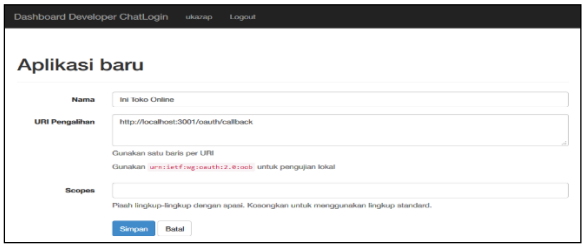


Gambar 1. Desain akhir: halaman beranda

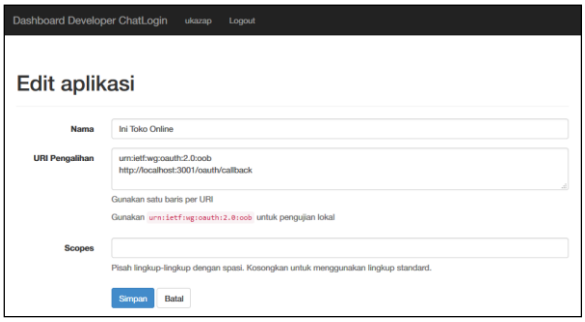
Gambar 2. Desain akhir: halaman registrasi



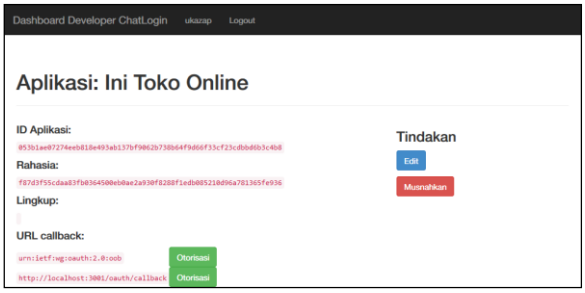
Gambar 3. Desain akhir: halaman *dashboard developer*



Gambar 4. Desain akhir: halaman dashboard developer, menambah aplikasi baru



Gambar 5. Desain akhir: halaman dashboard developer, mengedit aplikasi yang sudah terdaftar



Gambar 6. Desain akhir: halaman dashboard developer, info aplikasi

Gambar 7. Desain akhir: halaman login, pengguna melakukan input username

Gambar 8. Desain akhir: halaman login, provider menunggu konfirmasi dari pengguna.

Gambar 9. Desain akhir: Telegram, pesan konfirmasi dari provider

Dari penelitian ini diperoleh spesifikasi teknis, keunggulan, kelemahan, serta faktor-faktor dalam implementasi dari model akhir yang telah dikembangkan.

Identifikasi aktor dan *use case* secara *high level* yang terlibat dalam sistem yang akan dikembangkan adalah sebagai berikut:

Tabel 1. Identifikasi aktor

No.	Aktor	Deskripsi
1	*Provider*	Adalah pihak yang bertanggung jawab kepada pihak-pihak lain atas berjalannya layanan otentikasi serta melakukan administrasi dan moderasi.
2	*Developer*	Adalah klien yang memiliki aplikasi berbasis *web* maupun *mobile* yang mempercayakan otentikasi fungsi otentikasinya kepada *provider*.
3	Pengguna	Adalah pihak yang menyediakan identitasnya kepada provider untuk melakukan otentikasi ke dalam aplikasi yang dikembangkan oleh *developer*.

Berikut adalah beberapa fungsi-fungsi yang ada dalam sistem serta deskripsinya:

Tabel 2. Deskripsi use case

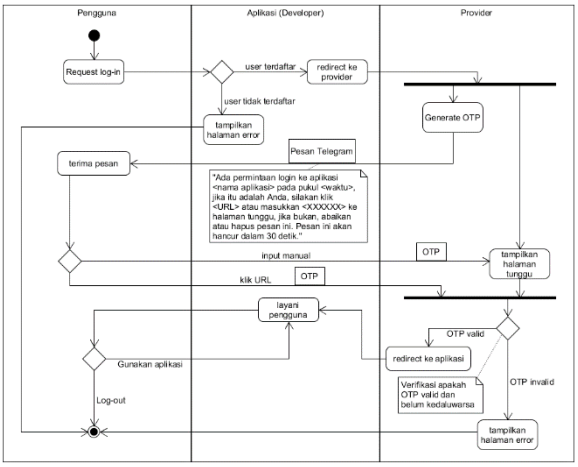
No.	*Use Case*	Deskripsi	Aktor
1	Mengelola *credentials*	Mencatat *developer* yang terdaftar, mencatat pengguna yang terdaftar, menerbitkan *access token* yang akan digunakan oleh *developer*.	*Provider*
2	Konfirmasi pengguna	Mengirimkan notifikasi kepada pengguna dan menerima konfirmasi permintaan otentikasi.	
3	Registrasi *developer*	Mendaftarkan diri sebagai *developer* untuk memperoleh hak mengelola aplikasi yang memerlukan layanan otentikasi.	*Developer*
4	Mengelola aplikasi	Menambahkan, menghapus, serta mengedit aplikasi-aplikasi yang akan digunakan oleh pengguna. Dari sini *developer* mendefinisikan nama aplikasi, *callback URL*, dan akan memperoleh *access token*.	

5	Akses API	Mengakses API untuk melaksanakan otentikasi pengguna ke dalam aplikasi yang dituju, ini memerlukan *access token*. Dilakukan setiap kali pengguna melakukan *login*.	
6	Registrasi pengguna	Pendaftaran diri untuk dapat melakukan otentikasi pada aplikasi-aplikasi yang terdaftar.	*Pengguna*
7	Otentikasi ke aplikasi	Login ke dalam aplikasi yang terdaftar, termasuk menerima dan membalas konfirmasi otentikasi.	

Perangkat lunak yang dibangun memiliki dua bagian utama yakni API untuk developer dan antarmuka pengguna. API yang dimaksud berbasis HTTP yang dapat digunakan oleh pengembang aplikasi untuk mendelegasikan otentikasi pengguna kepada provider, seperti halnya Facebook Connect. Antarmuka pengguna meliputi halaman depan provider yang mengenalkan pengguna maupun developer tentang layanan yang ditawarkan serta kelebihannya dibandingkan alternatif yang ada, halaman tunggu ketika pengguna mengirim request untuk login, dan dokumentasi/manual baik untuk pengguna akhir maupun untuk developer.

Kedua bagian tersebut dibangun menggunakan Ruby on Rails dengan data store PostgreSQL dan disajikan oleh web server NGINX. Pembangkitan OTP pengganti password dilakukan secara algoritmik menggunakan metode Time-based One-time Password untuk meminimalkan penggunaan ruang penyimpanan dan memberikan efek samping yang baik (yang tidak disengaja), yakni memungkinkan kompatibilitas dengan aplikasi Google Authenticator.

Rancangan provider berbasis OAuth2 dibuat menyerupai alur OpenID Connect. Alur otentikasi adalah berikut:



Berikut adalah penjelasan dari alur otentikasi di atas:

Tabel 3. Langkah-langkah alur otentikasi

No.	Aksi
1	Pengguna melakukan *request login* (mengisi *username* Telegram pada halaman *login* aplikasi)
2	Aplikasi memeriksa apakah *username* yang dimaksud terdaftar, jika terdaftar maka arahkan *(redirect)* pengguna ke halaman *provider*, sebaliknya jika tidak terdaftar tampilkan halaman *error*.
3	Provider menampilkan *halaman tunggu* selagi menciptakan *one-time password* menggunakan algoritma TOTP
4	Pengguna menerima pesan pada *Telegram Messenger* yang dikirim oleh *bot provider* yang berisi OTP, di sini pengguna dapat mengirim kembali OTP dengan dua cara

	yakni mengisi kode 6-digit pada *halaman tunggu* atau meng-klik tombol 'Izinkan' dari pesan tersebut. Tahap ini adalah dimulainya proses verifikasi di mana dilakukan pembuktian apakah *request login* benar-benar dikirim oleh pengguna yang berhak (pengguna yang sah memiliki akses pada akun Telegram-nya yang dapat menerima OTP).
5	Setelah menerima kembali OTP dari pengguna, *provider* memeriksa kembali OTP tersebut dengan algoritma TOTP dan apabila OTP sah dan belum melampaui masa berlaku (60 detik), maka arahkan pengguna kembali ke halaman aplikasi, sebaliknya tampilkan halaman *error* jika OTP tidak valid.
6	Pengguna dapat menggunakan aplikasi setelah diotentikasi oleh *provider*.
7	Aplikasi melayani pengguna hingga pengguna memutuskan untuk berhenti (*logout*).

Rancangan yang dihasilkan dan memiliki keunggulan dan kelemahan berikut:

A. Keunggulan

1. Mudah diintegrasikan dengan berbagai aplikasi pada berbagai platform (web/mobile/desktop) karena menggunakan standard OAuth2 yang berbasis protokol HTTP.
2. Pengiriman pesan yang singkat sebab menggunakan Telegram.
3. Pengiriman pesan tidak membutuhkan sebab tidak diperlukannya SMS gateway.
4. Mempermudah pengguna sebab tidak mewajibkan input kode secara manual pada halaman login.
5. Pengguna dapat melakukan login melalui berbagai perangkat (smartphone, tablet, komputer) dan berbagai platform (Android, iOS, Windows Phone, web, dan lain-lain).

B. Kelemahan

1. Tidak menjangkau pengguna perangkat telepon seluler yang tidak didukung oleh Telegram.
2. Pengguna harus mengaktifkan kunci pengaman pada smartphone.
3. Pengguna hanya bisa memasukkan username Telegram, login menggunakan e-mail belum didukung oleh provider.
4. Telegram belum sepopuler layanan lain seperti WhatsApp, BBM, Facebook Messenger, dll.

Selama pengembangan produk, ditemukan beberapa hal yang mendukung dan menghambat implementasi:

A. Faktor Pendukung

1. Semua komponen yang dibutuhkan tersedia secara cuma-cuma dan dilengkapi dokumentasi yang komprehensif, memudahkan dalam pemasangan.
2. Sudah ada standar yang jelas tentang otentikasi berbasis OAuth2/OpenID Connect.

B. Faktor Penghambat

1. Telegram Bot API tidak mengizinkan provider mengetahui nomor telepon yang digunakan oleh pengguna, sehingga sebagai alternatif digunakan chat_id untuk

identifikasi pengguna, dan karena itu pengguna diinstruksikan untuk mengatur username.

2. Meskipun Telegram memiliki jumlah pengguna yang banyak, ia masih kalah populer dengan aplikasi lain seperti WhatsApp, BBM, Facebook Messenger, dll.
3. Tidak banyak layanan web hosting yang mendukung deployment Ruby on Rails.

IV. DAFTAR PUSTAKA

Balthrop, Justin. (2014). *Passwords are Obsolete*. (Online), tersedia: https://medium.com/@ninjudd/passwords-are-obsolete-9ed56d483eb.

Houtven, L. Van. (2014). *Crypto 101*.

Krall, Jacob. (2013). *Beginner's Guide to TOTP*.

M'Raihi, D., Machani, S., Pei, M., & Rydell, J. (2011). *TOTP: Time-Based One-Time Password Algorithm*. Internet Engineering Task Force (IETF) RFC 6238, 1–17. (Online), tersedia: http://tools.ietf.org/html/rfc6238.

Rosetta Code. (2015). Time-based One-time Password Algorithm. (Online), tersedia: http://rosettacode.org/wiki/Time-based_One-time_Password_Algorithm, diunduh 20 Desember 2015.