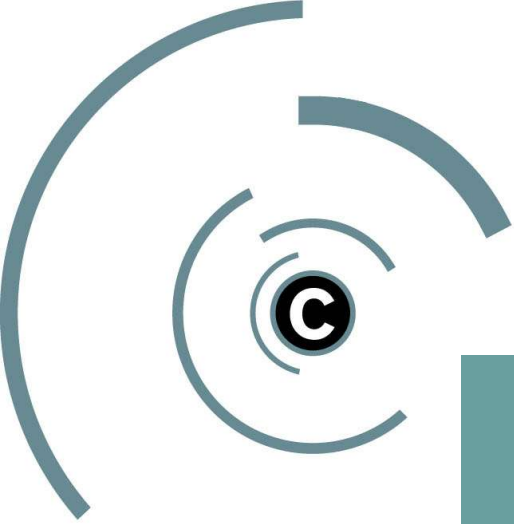


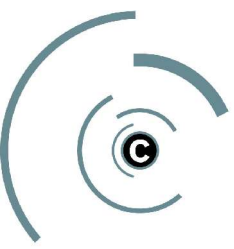


Deploying a TLS CA with OpenCA

A Real Life Use Case

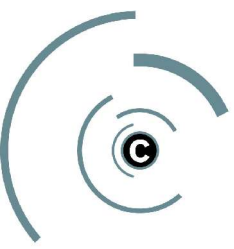
Dipl.-Ing. Martin Bartosch; Cynops GmbH
m.bartosch@cynops.de

2004-10-12



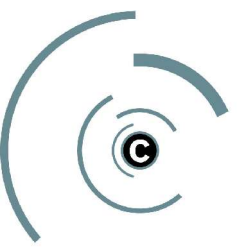
Contents

- Environment and requirements
- High level PKI design
- Implementation



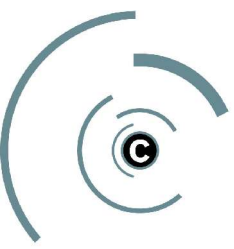
Environment

- Financial Institution
- Global scope
 - Intranet spread all over the world
 - Data centers in Europe, USA, Asia/Pac.
 - More than 2000 interconnected server nodes world wide
- High risk core business applications
 - Critical data exchanged across internal network
 - COTS Messaging Middleware
 - stunnel for generic TCP encryption



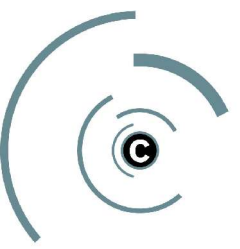
Requirements: Security

- End entities
 - Generic TLS server and client certificates
 - End-to-end security (SSL/TLS with client authentication)
 - PKCS#10 and “Basic Requests”
 - local scripts for communication with CA
 - CRLs via LDAP
- CAs
 - High security Root CA
 - Medium security Level 2 CA
 - CA private keys protected by HSMs



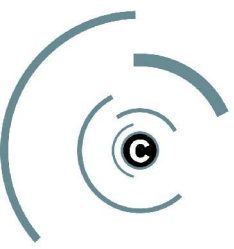
Requirements: Operating

- Initial certificate issuance < 3 hours
- Immediate certificate revocation and renewal
- CA availability: 24 hour downtime acceptable
- CRL availability: 24/7 required (LDAP only)
- Business continuity
 - Application availability is critical
 - Smooth migration
 - Ease of use for application owners
 - Automated processes



TLS CA: Processes and workflow

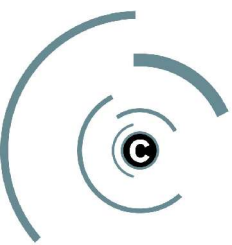
- Initial certificate request
 - Operating staff requests certificate
 - Application owner approves certificate
 - CA automatically issues certificate
 - Automatic certificate deployment
- Certificate renewal
 - Automatically scheduled 1 month prior to expiry
 - Certificate data verification



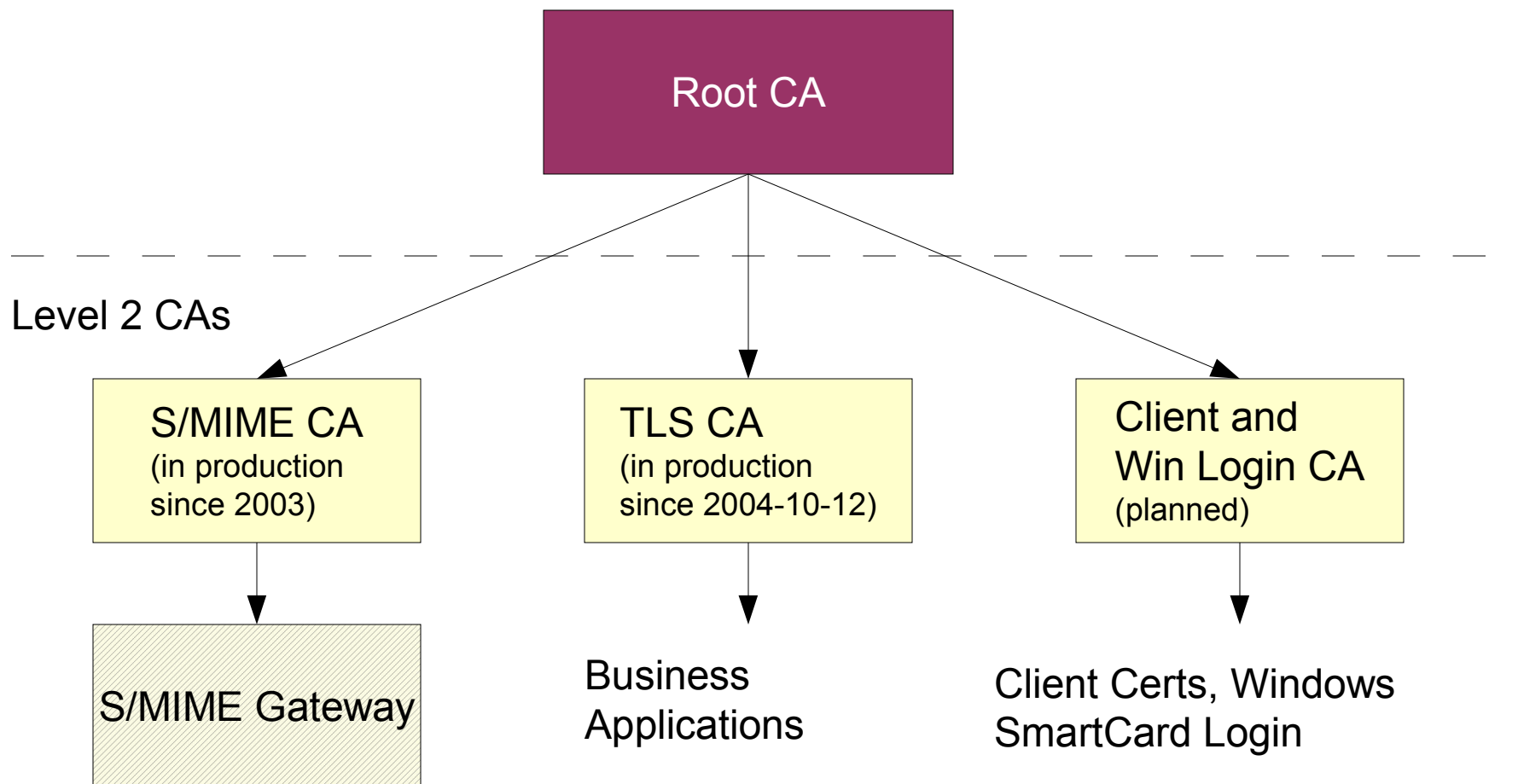
PKI Processes

PKI: 20 % Tech, 80 % Processes

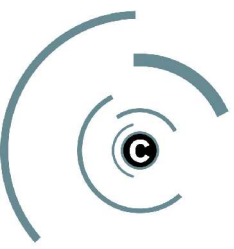
- policies, profiles
 - certificate profiles
 - CPS
 - PKI policy
- processes
 - “certificate life cycle management”
 - documentation
 - audit



PKI Architecture

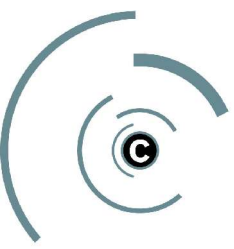


...and probably more to come



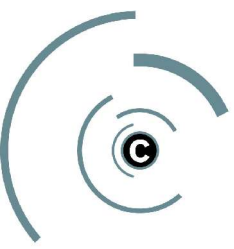
Design: Root CA

- Offline CA, components stored in a vault
- Private key protected by HSM
 - 2 nCipher nShield SCSI HSMs (production, backup)
 - 3/5 Administrator Quorum for administrative actions (e. g. HSM or SmartCard recovery)
 - 2/3 Operator Quorum for using the CA key
- Insight: a Root CA isn't really that complicated - thus no need for complicated software
 - Implementation: OpenSSL, Shell Script (and lots of error checking)
- **LOTS** of documentation



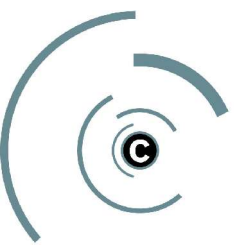
Design: TLS CA

- CA software must support business requirements
 - lot of custom features will have to be implemented
 - cost is an issue (it's not 2000 any longer...)
 - commercial PKI solutions lack flexibility
 - evaluated various Open Source solutions: IDXPKI, NewPKI, PyCa, OpenCA
- ▶ OpenCA was the winner...



Why OpenCA?

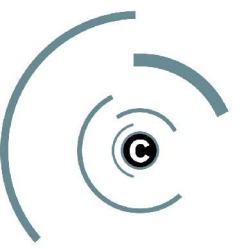
- Open Source Software
- Only Open Source CA with HSM support
- Large feature set
- More mature than competitors
- Modular design
- Implementation (Perl)
- Friendly developer community



OpenCA, extended

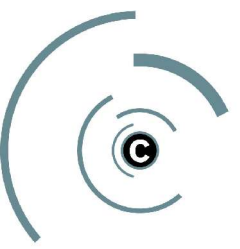
(or how I became an OpenCA developer...)

- nCipher HSM support
 - module/key online checks
 - error handling
- external authentication methods
 - username/password login utilizing existing frameworks (e. g. LDAP, proprietary systems)
- Finding errors, fixing bugs



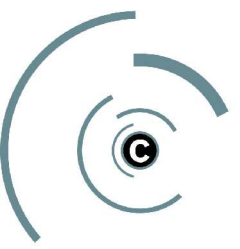
Design: TLS CA (2)

- CA private key protected by HSM
 - 3 nCipher nShield SCSI HSMs (2 production, 1 test)
 - SmartCard protection similar to Root CA
- OS/Platform: Linux on x86 Hardware
 - Other considerations: AIX, Solaris
 - x86 wins because of HW cost
- Software
 - OpenCA
 - Oracle DB
 - Tivoli system management
 - ADSM backup



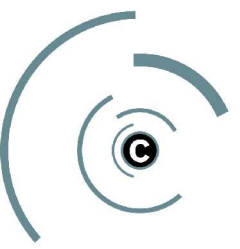
Design: TLS CA (3)

- System technology: configuration and deployment
 - System configuration stored in central deployment database
 - automatic installation on a blank machine takes 30 min
- Service availability
 - 24h CA downtime acceptable -> active/inactive setup
 - Production system periodically exports whole database to backup system
 - Transaction log exported real-time
- Increase availability according to demand
 - Migrate to SAN, HA cluster



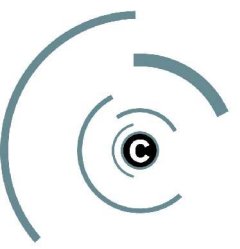
Design: TLS CA (4)

- Workflow initiated via web frontend
 - User authentication via LDAP or proprietary user authentication framework
 - Public frontend: any authenticated user
 - RA: authenticated business application owners
 - CA: delayed (30 min) automatic issuance of approved requests
 - CA manual frontend: CA operators



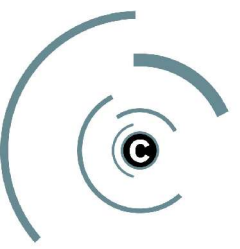
Design: TLS CA (5)

- Interfaces to existing infrastructure
 - Authentication framework deployed company wide (LDAP & proprietary)
 - CRL distribution to company LDAP
 - Future: automatic retrieval/verification of certificate request data using LDAP



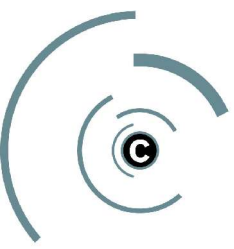
TLS CA: production use

- TLS CA is in pre-production test phase since August
 - successful interop tests with middleware infrastructure
 - test certs expire soon
- Production should start today
- Key ceremonies (Root and TLS) performed yesterday



TLS CA: the way ahead

- Agenda for the next months:
 - event auditing (session authentication information)
 - automatic Cert/CRL issuance and certificate renewal
 - CA key usage auditing
 - CA certificate rollover support



OpenCA TLS CA Use Case

Cynops GmbH – network security engineering

Dipl.-Ing. Martin Bartosch m.bartosch@cynops.de

Kirchgasse 10c

61449 Steinbach (Taunus)

fon 06171.6981803

fax 06171.6981809

mobile 0172.6614304