SECRET AS A SERVICE

Key Management for the Open Cloud
ABOUT US

ACADEMIC

DEVELOPER

SECURITY

CONSULTANT

APPLICATION

SECURITY

SECURITY

PRODUCTS

OWASP BOARD MEMBER

OWASP LIVE CD

OWASP WTE

RACKER SINCE ‘11

PRODUCT SECURITY

 HACKING THE RACK
Most important security technologies for a hoster to provide

- **Data Protection**: 73% (57% #1 Choice, 13% #2 Choice, 2% #3 Choice)
- **Endpoint & Network Protection**: 46% (19% #1 Choice, 11% #2 Choice, 16% #3 Choice)
- **Identity & Access Control**: 38% (11% #1 Choice, 18% #2 Choice, 9% #3 Choice)
- **Application Security**: 52% (7% #1 Choice, 27% #2 Choice, 18% #3 Choice)
- **Vulnerability & Incident Management**: 49% (4% #1 Choice, 18% #2 Choice, 27% #3 Choice)
- **Configuration & Patch Management**: 42% (2% #1 Choice, 13% #2 Choice, 27% #3 Choice)
Customers don’t want to give us their data

RISKY BUSINESS

Organization risk tolerance

- We are risk takers: 4%
- Somewhat willing to take risks: 30%
- About average: 21%
- Somewhat risk averse: 26%
- Risk averse: 19%

Cloud strategy regarding sensitive data

- We allow some sensitive data to reside in a public cloud: 31%
- We’re considering allowing some sensitive data to reside in a public cloud: 22%
- We don't/won't allow sensitive data to reside in a public cloud: 38%
- We don't have a public cloud strategy: 7%
- I don't know: 2%
Current Plans

PROTOCOL SUPPORT
Must support different protocols so that multiple products can integrate to the same system.

IDENTITY
Must support standard Keystone authentication methods.

MULTI-TENANT
Must support all tenants for a Cloud in the same system with guaranteed isolation.

AUDITING & COMPLIANCE
Must support auditing & logging to support various compliance regimes.

FREE & OPEN SOURCE
Must support for all environments, public and private.
Every OpenStack project has encryption needs

**FUTURE PLANS**

**CINDER, SWIFT & GLANCE**
Encrypted files at rest.

**RED DWARF**
Encrypted databases and tables.

**QUANTUM**
SSL Certificates and VPN keys.

**NOVA**
SSH keys, encrypted file systems.

**KEYSTONE**
Encrypted metadata, user level keys
Customer applications running on Cloud have a different, but overlapping, set of needs from OpenStack services.

**MULTI-CLOUD INTEROPERABILITY**
Customers want to be able to store their keying material in a different physical & legal environment than their data. We must support multi-cloud use cases and key sharing.

**EASY INTEGRATION**
Many legacy applications were not designed with advanced key management in mind. Customers need easy ways to retrofit existing applications, integrate new ones and connect vendor solutions.

**CENTRALLY MANAGED**
Key management is easy to get wrong. Customers need an easy to manage solution with optional expert assistance in configuration and monitoring.

**IMPROVED SECURITY & COMPLIANCE**
Most customers have compliance requirements to meet. We must support those needs while enabling real security improvements.
public class CryptHelper {
    private static final String ALOGRITHM = "PBEWithMD5AndTripleDES";

    // Salt
    static byte[] salt = {
        (byte) 0xc1, (byte) 0xa3, (byte) 0x28,
        (byte) 0x1c, (byte) 0x7b, (byte) 0xc9,
        (byte) 0x9e
    };

    static PBEKeySpec pbeKeySpec = new PBEKeySpec("chamber of secrets".toCharArray());

    public byte[] encrypt(String cleartext) { ... }
    public String decrypt(byte[] ciphertext) { ... }
}
# Be sure to restart your server when you modify this file.

# Your secret key for verifying the integrity of signed cookies.
# If you change this key, all old signed cookies will become invalid!
# Make sure the secret is at least 30 characters and all random,
# no regular words or you'll be exposed to dictionary attacks.

```ruby
::Application.config.secret_token = "{{SECRET_TOKEN}}"
```
Bad Advice

First 10 results on Google - all bad

BADADVICE

HARDCODED KEYS

HARDCODED ALGORITHMS

BAD CYPFER BLOCK TYPES

OLD

NULL & HARDCODED IV’S
Open source key management

INTRODUCING CLOUDKEEP

https://github.com/cloudkeep
Barbican is the main ReST API providing secret storage, provisioning, lifecycle management auditing and reporting. It is written in Python using Falcon, Oslo and following OpenStack standards.

Postern is the agent that provides access to secret material. It is currently planned to be implemented in Go, but we may use the Rackspace Cloud Monitoring agent framework (Virgo).

Palisade is a client side JavaScript MVC application that provides a web interface for Barbican. It is written in AngularJS and can be delivered from the Barbican API server or as a Chrome plugin.

Keep is a python based command line client similar to python-novaclient. It is most useful for server maintenance, troubleshooting and development.
1. Provide a central key-store capable of distributing keying material to all types of deployments including ephemeral Cloud instances.
2. Support reasonable compliance regimes through reporting and auditability.
3. Application adoption costs should be minimal or non-existent.
4. Build a community and ecosystem by being open-source and extensible.
5. Improve security through sane defaults and centralized management of key policies.
6. Out of band communication mechanism to notify and protect sensitive assets.
7. Use OpenStack tools, processes, libraries and design patterns to ensure easy integration into the ecosystem.
ATTACK OF THE PLUGINS

Hardware Security Modules (HSM)

Cloud Load Balancers

Automatic Provisioning Targets

API

Internal & External Certificate Authorities

Data Storage Backends
The agent presents a FUSE file system to allow applications easy integration options.

Enforces Policies
Each secret has a set of policies that dictate its use. These policies are mostly enforced by the agent.

Keystone Integrated
The agent uses keystone for identity, pairing and policy management.

Out of Band Communication
The agent communicates with the API to represent real-time data about secret usage.
Reusable block of configuration governing secret access

**EXAMPLE POLICY**

```json
{
    "uuid": "01fb57ff-058c-4d68-...",
    "name": "Polyglot policy",
    "max_key_accesses": 1,
    "minutes_available_after_reboot": 10,
    "events": {
        "log_sinks": ["api", "syslog"],
        "file_path": "/var/log/postern.log",
        "allow_override": false,
        "allow_panic": true,
    },
    "executable": {
        "minutes_available_after_restart": 10,
        "name": "mysqld",
    },
    "filesystem": {
        "directory_name": "chamber",
        "owner": "root",
        "group": "root",
        "listable": false,
    },
    "path": "/usr/bin",
    "owner": "mysql",
    "group": "mysql",
    "hash": "44aea8f32fa3f1f4..."
}
```
The keying material

**EXAMPLE SECRET**

```json
{
    "uuid": "e2b633c7-fda5-…",
    "cacheable": false,
    "expiration": "2014-02-28T19:14:44.180394",
    "secret": "b7990b786ee9659b43ec5…",
    "secret_type": "application/aes-256-cbc",

    "filesystem": [
        {
            "name": "configuration_key",
            "presentation": "file",
            "permissions": "300",
            "owner": "root",
            "group": "root"
        }
    ]
}
```
DEMO TIME
QUESTIONS?