I am not sure how some of these thoughts relate to the workshop's limited scope in terms of Web Server Certificates. But, here are some of the thoughts I have had. May be they help as lessons learned as we move forward and give us ideas on what to do and what not to do.

1 What Keeps Me Awake at Night

Someone will hack into the CA give themselves a self-issued certificate and delete it from CA database and thus will be undetected unless accidently stumbled on the usage side.

Someone will hack into the CA, exploit side channel (e.g., timing channel), get the CA key and cover their tracks.

Lack of rigor in primality tests for the keys in the HSM may lead to low work factor to discover the CA key.

2 If I Were to Do X.509 Again

2.1 Self-Issued and Path Length

We have publishes pitfalls with self-issued certificates at a prior NIST PKI Research Workshop. Now in the light of Mozilla request for certificate scan, one feels that the path length constraints is not particularly helpful. Specifically, an attacker will get itself issued a self-issue certificate. If I were to redo X.509, I would apply all of the constraints (or at least the path length constraint) to the self-issued certificate.

It should be noted that operationally one generally does not need self-issued certificates. While a case can be made for their usefulness in the case of trust anchors, other mechanisms such as TAMP; SET approach to release the hash of the next key; NSA approach of releasing encrypted next key; etc. can be helpful. Finally, even if self-issued is used for trust anchors, given they generally should and do contain no path length constraints, not counting self-issued certificates in path length was an unfortunate and erroneous decision.

2.2 CP/CPS

I would not use the word "policy" for the extension in the certificate and for the CP/CPS document. The word "policy" has a meaning in this town and this town is full of "policy wonks", good and bad.

My favorite quote: "You are an engineer. What are you doing looking at a CP/CPS".

Most CPS I see are worthless and should not have been written except for statement that we abide by the CP. (Since this is all they say anyway). A CPS should give you a clear picture of how the system works, what the procedures are, and how the security controls are implemented. This allows the reader to understand and analyze the security of the system. In short, the CP/CPS and compliance audit discipline has been an utter failure and mostly waste of time. It has become one of those endeavors where 90% of the time is spent in getting 10% benefit and that too at the expense of more critical (from security and from interoperability viewpoints) 90% of the work.

2.3 EKU for the Path

Microsoft had a good idea. EKU should be enforced on the path, just like name constraints and policies. This approach allows us to segregate CAs for high value certificates such as code signing and time stamp service. The approach also allows to constrain cross-certification based on EKU, i.e., applications. For example, one organization may not wish to trust web servers or code signing or time stamp authority from another.

This can still be achieved by defining specific EKU and associating appropriate semantics with them or by defining another extension. Whether and when the application support comes is the issue.

2.4 Crypto Binding

The X.509, OCSP Standards and applicable RFC should call for crypto binding between certificate and associated revocation provider. It provides simplicity in terms of reduced computational complexity and reduced software complexity. It also provides high degree of security via crypto binding. This concept was explored in depth in a research paper we submitted to the NIST PKI Research Conference circa 2005.

To further expound on this axiom, the CRL should be signed (and more critically must be only verified by the relying parties) using the same key that the certificate in question was signed with. Microsoft used to (any possibly still does, bit I am not sure if that has changed in Windows 7) enforce this. Similarly, RFC 2560 implies and most OCSP Responders serve OCSP responses that are verifiable using the certificate signed using the same CA public key as the certificate in question. It is critical that OCSP clients enforce this constraint to enhance security and to reduce both computational and software complexity. Our attempts to codify this as a part of client side processing for RFC 2560 were not successful. NIST edited a version of the RFC. Our compromise to accommodate NIST editor and either enforce the crypto binding or enforce the trust models (in order to shore up the security) at the client side also fell on deaf ears.

One of the lessons from this effort is detailed client/user/relying party processing rules are critical to preserving security. NIST and IETF should take note of this axiom.

3 False Sense of Security

3.1 Not Issued Per OCSP

The CAB Forum has mandated that an OCSP Responder must inform the CA if the OCSP Responder detects a certificate that has not been issued by the CA. To take that further, the IETF PKIX working group wasted inordinate amount of time arguing over this feature which has limited value.

From these debates and CABF forum requirement one wonders if folks are fully understanding and analyzing the threats. One should not rely on such weak mechanism and one should not waste inordinate time in perfecting a weak mechanism.

Note that the mechanism is weak because if the CA is compromised, a half-way intelligent attacker will either mint an OCSP Responder certificate as well and mint certificates pointing to the rogue Responder or will mint certificates that omit OCSP field in the AIA extension altogether.

3.2 Certificate Scan

Mozilla has come with additional and another requirement for certificate scan. Again, an attacker would cover their tracks and such exercises will prove futile. Making such requirement mandatory without considering the effectiveness of the mechanism smacks of lack of analysis or knee jerk reaction. What will be useful is to have proper controls on the issuance mechanisms such as template, profiles and policies and have mechanisms to check applicable mechanisms and data to protect against accidental mistake or against misconfiguration. Also, see additional recommendations for checking of RSA exponent, key size, and key quality elsewhere. It is recommended these features not be examined as part of certificate scan, but be part of checks during the issuance process.

4 Will it Work

4.1 Certificate Transparency

The recent work on Certificate Transparency (CT) offers an avenue for detection of misissued or unauthorized issued certificates. But, the effort raises couple of questions.

1. The details of the client side behavior are not fully defined and use of the term “should” mean that some of the client side security critical requirements (i.e. processing rules) need not be enforced.

2. What makes one think that browser vendors who have not implemented stable X.509 features will implement the new capability.

3. The effectiveness of CT depends on human-error prone and labor intensive effort to examine the log to discern unauthorized issuance.

5 Some Wrong Advise

5.1 EKU in Federal PKI

FPKI seems to be leaning towards not asserting EKU in certificates that have Digital Signature bit set. In MSFT Windows environment such certificates can act as legitimate code signer, resulting in undesirable exposure due to human mischief, human error or attack on less protected client workstations. The FPKI position seems to be driven by NIST advice. NIST advice seems to be driven by interoperability. There are several inconsistencies in this position and associated rationale:

1. It is erroneous to assume that the PKI must provide interoperability and optimize security. PKI is an infrastructure for security services. Thus, security is a must and the goal must be to optimize interoperability while not sacrificing security.

2. The assumption that there software developers out there building software for Federal use and are requiring OIDs that are not widely known is not supported by empirical research.

3. The assumption that even if such was the case, these developers are no doing the same thing for security critical usage as MSFT has done for code signing and hence further exposing the relying party via absence of EKU is equally facetious.

4. Note that even as obscure EKU OIDs come to light, they can be asserted in newly issued certificates and hence we can continue to increase and enhance interoperability.

6 Some Sound Advise

6.1 Issuance Checks

Rather than doing certificate scans, the CA vendors should implement the following controls which are easily testable and verifiable. These include:

1. Verification that the RSA exponent complies with FIPS 186-3. For example, it is not unusual to see an exponent of 3 in some OCSP certificates.

2. Verification that the public key meets the key size requirement (e.g., 2048 RSA modulus, P256 EC etc.)

3. Verification of the validity of the public key as described in references cited in Section 5.4.3 of NIST SP 800-57 Part 1.

6.2 RSA Key Generation

Generally, none of the FIPS 140-2 validated cryptographic modules have been validated to comply with key generation requirements for RSA keys in accordance with FIPS 186-3. These include cryptographic modules used by the infrastructure components such as the CA, CMS, OCSP etc. and the end-user tokens such as the PIV and PIVI cards. Security concerns fall into three broad categories:

1. Insufficient primality tests resulting higher probability that the one of the factors of the modulus is not a prime.

2. Randomness of the factors

3. Entropy for the random

6.3 Randomness in Certificate

Browser software vendors are beginning to require 32 bits of randomness in the serial number in the certificates. The primary motivation for this is to mitigate the hash collision threat. Randomness in the serial number can constrain how one designs and implements an OCSP solution. A perfectly fine alternative would be as follows:

1. Not require randomness when stronger hash such as SHA-224 or stronger is used

2. Not require randomness when the certificate request is crafted and securely submitted by a trusted entity such as a CA, RA or OCSP Responder.

3. Permit randomness in other fields such as subject DN, validity period.

4. Permit only 20 bits of randomness since the probabilistic work factor involved in overcoming even 20 bits of randomness is very high/

6.4 Device Certificates

There seems to be a rush towards issuing certificates for devices for device authentication. Technology being proposed is that of auto issuance such that if the LAN or Enclave is compromised, the CA can be either compromised or can be made to issue certificates of the attacker's choice.

The concept of device certificates actually dilutes security since this weaker infrastructure can be used to mint certificates that can be used as say PIV or PIVI certificates.

Concepts such as policy separation will not be effective since most relying party software do not implement sufficiently rich policy processing capabilities.

Concepts such as name constraints will have limited effect if and only if devices name spaces can be made distinct from those for other higher assurance certificate subjects. Even then, the effectiveness will come at access control and not authentication level, effectively eroding one layer of security.

Concepts such as distinct root for devices will not work for the machines and applications that require the device root as well as root for higher assurance PKI.

Concepts such as certificate usage metadata in MSFT Windows environment may reduce, but not eliminate the exposure in Windows environment once device roots or device CAs are limited to a few purposes. Further research is required in this area.

In light of this, one wonders why not use a weak non-PKI mechanism (note that for devices PKI would be a weak mechanism anyway due to the issuance process and tight binding between the device and CA) to perform device authentication. There are several non-PKI mechanisms available in the commercial space. The approach will have significant cost saving and will close a potential attack vector against the high-assurance PIV and PIVI PKI.

6.5 CA Trust Research

There are several approaches that can be taken to mitigate the threats of CA and/or RA compromise and these need to be studied further to determine which mechanisms and combinations thereof provide cost-effectiveness and scalability while preserving security:

These include:

1. Multiple signatures from multiple CAs under a single signature algorithm OID
2. Split signatures from multiple CAs
3. Multiple certification paths
4. Certificate Transparency
5. SCVP
6. CA Broker
7. Trusted Time Stamp by CA Broker
8. Probabilistic Trust in CA by extending concepts of perception/convergence

7 Summary

In summary, the following observations are made. They are in no particular order of importance.

1. Work with cryptographic module vendors to accelerate the following:
    a) Mitigate side channel attacks on cryptographic modules used by the infrastructure components such as the CA, CMS, OCSP Responder etc.
    b) Have module certified for compliance with FIPS 186-3 for key generation.
2. Revise the standards and have PKE software implementers' so that self-issued certificates count in path length constraint,
3. Always develop standards which define relying party/consuming party processing rules completely.
4. Sponsor research and analysis on CA trust to mitigate the exposure due to CA compromise.

5. Get rid of CP, CPS and compliance audit discipline and replace with documenting system details, security analysis, penetration testing and monitoring.

6. Define EKU OIDs for certification path in order to constrain the CAs and users for appropriate applications

7. Require CAs to implement the NIST SP recommendations in the area of public key validation

8. Require CAs to implement policies related to algorithms and key sizes.