Locally-Centralized Certificate Validation and Its Application in Desktop Virtualization Systems

Bingyu Li, Jingqiang Lin, *Senior Member, IEEE*, Qiongxiao Wang, Ze Wang, and Jiwu Jing, *Member, IEEE*

***Abstract*— To validate a certificate, a user needs to install the certificate of the root certification authority (CA) and download the certificate revocation information (CRI). Although operating systems and browsers manage the certificate trust list (CTL) of publicly-trusted root CAs for global users, locally-trusted root CAs still play an important role and it is difficult for a user to manage its CTL properly by itself. Meanwhile, the CRI access is inefficient, sometimes even unavailable, and causes privacy leakage. We revisit these problems by analyzing the TLS sessions within an organization. To the best of our knowledge, we are the first to analyze CTL management and CRI access on the scale of medium-sized organizations. Based on the analysis, a *locally-centralized* design is proposed to manage the CTLs of all users by IT administrators and access the CRI services for all users, within an organization. We apply this design to desktop virtualization systems to demonstrate its applicability, and build *vCertGuard* with oVirt and KVM-QEMU. In vCertGuard, the CTLs of all virtual machines (VMs) are managed in the VM monitors (VMMs). In the CTL, the self-signed certificates of publicly-trusted root CAs are properly configured, while each locally-trusted certificate chain is specified one by one. vCertGuard accesses the CRI services for all VMs, and the downloaded CRI is cached and shared among VMs. Because most TLS servers are visited by multiple users of an organization, it reduces the cost of CRI access. Experimental results of the prototype system show that vCertGuard maintains the CTLs with a negligible overhead, and significantly improves the performance of CRI access.**

***Index Terms*— Public key infrastructure, trust management, certificate revocation, virtualization-based security.**

Manuscript received February 14, 2020; revised August 4, 2020 and September 6, 2020; accepted October 3, 2020. Date of publication November 2, 2020; date of current version December 1, 2020. This work was supported in part by the National Key Research and Development Plan of China under Grant 2017YFB0802100, in part by the National Natural Science Foundation of China under Grant 61772518 and Grant 62002011, and in part by the Open Project of State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, under Grant 2020-ZD-05. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Marina Blanton. *(Corresponding author: Jingqiang Lin.)*

Bingyu Li is with the School of Cyber Science and Technology, Beihang University (BUAA), Beijing 100191, China, and also with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China (e-mail: libingyu@buaa.edu.cn).

Jingqiang Lin is with the School of Cyber Security, University of Science and Technology of China, Hefei 230026, China (e-mail: linjq@ustc.edu.cn).

Qiongxiao Wang is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China, also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: wangqiongxiao@iie.ac.cn).

Ze Wang is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China (e-mail: richardwangze@foxmail.com).

Jiwu Jing is with the School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: jwjing@ucas.ac.cn).

Digital Object Identifier 10.1109/TIFS.2020.3035265

I. Introduction

PUBLIC key infrastructures (PKIs) provide services through certificates. Certificate-based authentication, confidentiality and data integrity are designed in TLS. To validate a TLS server certificate, a client needs to (*a*) install the self-signed certificate of the root certification authority (CA) and (*b*) download the certificate revocation information (CRI) [1].

It is difficult to manage the certificate trust list (CTL) of root CAs properly. The vendors of operating systems (OSes) and browsers evaluate the certification practice of CAs and maintain the public CTLs for global users. These self-signed certificates of *publicly-trusted* root CAs are by default installed in OSes or browsers. Because not all publicly-trusted CAs certify TLS servers [2], it expands the attack surface due to the security incidents of some CAs [3], [4]. Meanwhile, an organization may establish a root CA by itself, trusted only by its employees and users. Such *locally-trusted* root CAs are widely adopted [5], [6], but end users usually do not correctly understand the effects of installing a root-CA certificate [7], [8] and cannot properly manage the CTLs.

It is inefficient for browsers to access CRI services, when validating a certificate. The popular services of certificate revocation list (CRL) and online certificate status protocol (OCSP) involve extra network connections to other servers when a browser validates the TLS server certificate.[1] CRL files may grow to be megabytes in size, and sometimes even more than 70M bytes [10], [11]. Moreover, some issues prevent PKI clients from obtaining CRI: (*a*) CRI servers may be temporarily unaccessible due to network errors or attacks [12]; (*b*) special VPNs or proxies are needed to access some CRI services, and some might be even unavailable [13]–[15]; and (*c*) the certificates signed by locally-trusted CAs are usually not configured with any CRI distribution point (DP) [5]. Finally, privacy leakage happens when browsers access CRI services. An OCSP request includes the identifier of the validated certificate, so the OCSP server knows the website that the client is visiting [10], [12].

In this paper, we revisit these problems within an organization, but not on the scale of the Internet. We analyze more than two million TLS handshakes by 218 users of our department in the daily works of over one month, and finish the analysis:

[1]OCSP stapling [9] addresses this limitation by allowing the TLS server to staple OCSP responses in the TLS handshake, but it is not widely adopted [10], which is confirmed in our study.

Only a Small Number of Publicly-Trusted Root CAs Are Involved, While a Great Number of Locally-Trusted CAs Are Done: In the collected TLS sessions, 3,198 unique certificate chains are observed and each appears several times. 3,080 certificate chains are signed by 40 publicly-trusted root CAs, while 118 locally-trusted certificate chains are signed by 110 locally-trusted root CAs. Almost every locally-trusted root CA signs only one certificate. When publicly-trusted certificate chains appear in much more TLS sessions than locally-trusted ones (545,437 vs. 2,746),[2] from the point view of CTL management, locally-trusted root CAs play a more important role (40 vs. 110).

The CRI Access for Publicly-Trusted Certificates Is Inefficient, and the CRI Services of Locally-Trusted Ones Are Usually Unavailable: 2 out of 292 CRL DPs and 5 out of 106 OCSP servers of publicly-trusted certificates are unaccessible through normal networks. Some are accessible only through encryption proxies (i.e., Shadowsocks in the experiments). From the accessible CRI DPs of publicly-trusted certificates, the average time of CRL download is 1,512 ms (the maximum is 13,127 ms, and 36% takes more than 1,000 ms), and the average of OCSP responses is 411 ms, with the maximum of 1,438 ms. Meanwhile, 61% of the locally-trusted certificates are not configured with any CRI DP, and most CRI DPs configured in these certificates are invalid or unaccessible.

A CRL File (or OCSP Response) Is Likely to Be Needed by Different Users for Multiple Times During Its Validity Period: We extracted 3,187 OCSP DPs and 305 CRL DPs from all certificates. Among the 3,072 OCSP DPs for TLS server certificates, 23% are needed by multiple users and 40% are done for more than one times, within the typical validity period of 4 days. As for the OCSP DPs for intermediate CAs, the numbers increase to 61% and 74%, respectively, within the typical validity period of 7 days. A CRL DP includes the revocation status of multiple certificates, so each CRL file is needed by more users for more times. Among the 221 CRL DPs extracted from server certificates, within the typical validity period of 7 days, 40% are needed by multiple users and 59% are for multiple times. As for the CRL files of CAs, within the typical validity period, the numbers increase sharply to 94% and 95%, respectively.

Based on these observations, a *locally-centralized* design is proposed to manage the CTLs of all users, and to access the CRI services for all users, *within an organization*. All CTLs are configured by IT administrators that understand the security requirements, and the downloaded CRI is cached and shared among all users. The design brings these advantages:

It Simplifies and Secures CTL Management: A desktop user does not need to manage his/her CTL. Moreover, trusted certificates are managed at different levels of granularity in the CTLs: the self-signed certificates of necessary publicly-trusted root CAs are listed, while locally-trusted certificates are specified by each certificate chain (but not root CA certificate). This asymmetric design eliminates the potential threat of fraudulent certificates signed by locally-trusted CAs, for a locally-trusted CA is usually protected much worse than publicly-trusted CAs. It does not cause extra efforts because almost all locally-trusted certificate chains are signed by different root CAs.

The Performance of CRI Access Is Greatly Improved, and the Privacy Leakage Is Mitigated: Because most CRI is needed by multiple users of an organization, as shown in our analysis of the data collected, the cached CRI is likely to be queried by different users, and then the cost of CRI access is greatly reduced by CRI caching. Moreover, less privacy is leaked to CRI servers because the CRI servers could only trace the access of a group of users, but not a specified user.

We apply this locally-centralized design to desktop virtualization systems, and build *vCertGuard* to demonstrate its applicability and evaluate the performance. Desktop virtualization is a popular cloud-based desktop computer solution for a group of users. With the help of desktop virtualization systems, vCertGuard maintains the resources for certificate validation in the VM monitors (VMMs). VMMs provide services in a centralized way for all users (or virtual desktop computers). vCertGuard provides read-only CTL and CRI query services to VMs as virtual devices, and a VM accesses the vCertGuard virtual device to return its CTL or obtain CRI identified by the CRI DP. We define a CRI DP of CRL as the URL of the CRL file, and a CRI DP of OCSP is the URL of the OCSP server with a certificate identifier.

Building vCertGuard in desktop virtualization systems brings the extra benefits of isolation and efficiency as follows. Firstly, vCertGuard provides read-only query services to VMs. Only IT administrators are able to add/remove a certificate (chain) to/from the CTLs in the VMMs, and even an attacker completely compromising a VM could not install fake root-CA certificates in the CTL. It also masks the CRI DPs unaccessible to VMs, through special VPNs and proxies in the VMM. Secondly, the invocation of vCertGuard services from a VM is very efficient. Even when some CRI is uncached and accessed for the first time, the extra communication overhead between a VM and the VMM is negligible, compared with the cost of CRI download from the Internet.

We implement the vCertGuard prototype with oVirt [16] and KVM-QEMU [17], [18], and evaluate it by invoking the services in Chromium to validate TLS certificates. Experimental results demonstrate that (*a*) the overhead of the vCertGuard CTL service is negligible, and (*b*) the vCertGuard CRI service reduces the time cost of OCSP and CRL access by 90.1 - 98.4% and 75.0 - 98.1%, respectively, in different scenarios, compared with a "perfect" browser that implements revocation checks for each server certificate in TLS handshakes.

In summary, vCertGuard provides certificate validation with the following enhancements:

- Professional. The CTLs of all end users are configured by IT administrators, and all CRI needed is automatically accessed, cached, and updated. This effectively reduces end users' efforts and achieves professional PKI resource management services.
- Secure. The trusted certificates are managed at different levels of granularity: the locally-trusted certificates are not completely trusted as the global ones, but specified by

[2]Due to session reuse, a certificate does not appear in every TLS handshake.

each certificate chain in the CTLs. This design mitigates the potential threat of fraudulent certificates signed by locally-trusted CAs.

- Privacy-preserving. The CRI requests are sent on behalf of all users within an organization, and thus the CRI servers could only trace the CRI access of a group of users, but not a specified user. This protects a user's profile of network visits.
- Efficient. The downloaded CRI is cached and shared among all users, to improve the accessibility of CRI and the performance of CRI access. Because most CRI is needed by multiple users of an organization, the cost of CRI access is greatly reduced by CRI caching.

Our contributions are as follows. To the best of our knowledge, this is the first time to analyze the requirements of root-CA certificates and revocation checks on the scale of a medium-sized organization. Existing studies on TLS certificates provide various views on the scale of the Internet, but do not focus on the similarity of different users within an organization. Based on the analysis results we design a locally-centralized certificate validation solution, and apply the design to implement vCertGuard in desktop virtualization systems.

The remainder of this paper is organized as follows. Section II analyzes the certificates received within an organization. Section III presents the locally-centralized design and the implementation of vCertGuard. vCertGuard is evaluated in Section IV. Section V surveys related works and Section VI draws the conclusions.

II. Certificates Validated Within an Organization

In this section, we describe the collected dataset and present the detailed analysis results. Then, we discuss whether these analysis results still hold in other organizations, by comparing our dataset with other data collections.

A. Data Collection

We passively monitored the network gateway of our department, and collected TLS handshake packets from July 11, 2017 to August 12, 2017. The desktop computers of this data collection are connected through a 100Mbps Ethernet in one building, and the uplink port is 1Gbps. It consists of 2,524,937 TLS sessions of 224 users, distinguished by IPv4 addresses. Our department statically assigns IPv4 addresses to employees and students, usually configured on their desktop computers. Some may use laptops through the Ethernet with the assigned addresses, or through WIFI with dynamical IPv4 addresses. The dataset does not include any IP addresses dynamically-assigned for WIFI access.

After reviewing the dataset, we filtered out 6 users who scanned TLS servers on the Internet for their own research. The scan behaviors were confirmed by the users. The final dataset consists of 2,234,277 TLS handshakes collected from 218 users (or IP addresses). There are totally 3,198 unique certificate chains in 548,183 sessions, and about 75% of TLS handshakes without any certificate due to session reuse.

TABLE I
Analysis Results of Certificate Chains

	C_{cc}	C_{ts}	C_{ur}	C_{un}
Publicly-trusted certificate chain	3,080	545,437	40	3,202
Locally-trusted certificate chain	118	2,746	110†	44

C_{cc}: the number of unique certificate chains;
C_{ts}: the number of TLS sessions where the certificate chains appear;
C_{ur}: the number of unique root CAs with the certificate chains;
C_{un}: the number of unique non-self-signed certificates.
†78 TLS server certificates are self-signed.

TABLE II
Analysis Results of 110 Locally-Trusted Root CAs

	Type	C_{lc}	C_{ts}	C_{ut}
Internal	Service website	5	634	102
	Device administration/configuration	15	297	7
	Router and cable modem	16	493	39
	Firewall	10	362	11
	VPN	7	194	19
	Other device (IP camera, printer, etc.)	8	75	13
External	Service website	39	264	56
	End device	10	427	7

C_{lc}: the number of unique locally-trusted root CAs;
C_{ts}: the number of TLS sessions where locally-trusted root CAs appear;
C_{ut}: the number of users that need to trust locally-trusted root CAs.

B. Analysis Result on Certificate

In the 3,198 unique certificate chains, 3,080 publicly-trusted certificate chains are signed by 40 root CAs and 118 locally-trusted ones signed by 110 root CAs. We use the Censys service [19] to judge whether a certificate is publicly-trusted or not, i.e., signed by a root CA trusted by Windows, Mozilla NSS, Apple iOS or OS X. Details are listed in Table I.

Publicly-trusted certificates appear in 545,437 TLS sessions and each one appears 177 times averagely during the collection. We analyzed the publicly-trusted root CAs in the TLS handshakes by each user, and found that each user needs to trust only on average 12 publicly-trusted root CAs. However, hundreds of root-CA certificates are installed by default; e.g., 361 root-CA certificates in Windows, 155 in Mozilla NSS, 173 in Apple iOS and OS X, and the union consists of 382 certificates [20]–[22]. About 90% of root CAs in these public CTLs are not involved in any TLS sessions, so it is imperative to remove some publicly-trusted CA certificates to mitigate the threat due to intruded CAs [2], [23], [24].

From the point view of CTL management, more attention shall be paid to locally-trusted certificates than publicly-trusted ones, because much more locally-trusted root CAs are involved (40 vs. 110). We examined the 118 locally-trusted certificate chains and found no attack. The purposes of locally-trusted certificates have been analyzed [5], [25]: most are signed for internal/external websites. In addition, there are lots of locally-trusted certificates for end devices or network devices, such as IPTVs, IP phones, firewalls, routers, VPNs, and remote storage devices. These purposes are confirmed in our study, as listed in Table II: among the 110 locally-trusted root CAs, 61 are used in information systems or network devices of our department (5 internal websites, and 56 network devices), and 49 are to access external websites or network devices.

TABLE III
ANALYSIS RESULTS OF CRI DPs IN THE CERTIFICATES

	C_{un}	w/ CRL		w/ OCSP	w/o CRI
		R_{un}	R_{dp}	O_{un}	I_{un}
Publicly-trusted	3,202	2,779	292	3,175	0
Locally-trusted	44	16	13	12	27

C_{un}: the number of unique non-self-signed certificates;
R_{un}: the number of unique certificates with CRL DPs;
R_{dp}: the number of unique CRL DPs;
O_{un}: the number of unique certificates with OCSP DPs;
I_{un}: the number of unique certificates without any CRI DP.

Among the 61 locally-trusted root CAs managed by our department (i.e., by the same group of IT administrators), 58 are at the same physical location as the users (including 2 internal websites and 56 network devices), and the other 3 internal websites are located at different buildings. Among the 49 locally-trusted root CAs for external websites or devices, 17 are managed by 14 third parties that provide public services including 6 universities, and 8 public agencies for tax, ticket booking, etc. The remaining 32 are used for remote administration and access to external end devices.

In our dataset, 154 locally-trusted certificates including root-CA certificates and non-self-signed certificates, account for only 5% of collected certificates, much less than the result of 87.9% in previous studies [5]. The major reason for this difference is that, their datasets come from the whole-Internet 443-port active scan [5] but we focus on the TLS certificates in daily works and our dataset filters out the TLS server scans. If the certificates scanned by the 6 users are counted, which probably do not cover the whole Internet, the proportion of locally-trusted certificates will be 33%. According to the 230 *discrete scans* [5], about 60% of locally-trusted certificates are ephemeral, i.e., appear in only one scan of the 222-week IPv4 scans. On the contrary, in our 33-day *continuous monitoring* of daily works, most locally-trusted certificates appear in hundreds of TLS sessions of several users.

C. Analysis Result on CRI

We extracted 3,125 CRL DPs from the certificates with CRL DPs (2,779 publicly-trusted certificates and 16 locally-trusted ones), and there are 305 unique CRL DPs. In these certificates, 326 are configured with two CRL DPs and 2 are with three, all of which are intermediate-CA certificates. There are 23 unique alternative CRL DPs in these certificates: 20 appear always as alternative DPs, and 3 also as the preferred (or first) CRL DPs in others. Totally 3,187 unique certificates are configured with OCSP DPs of 113 OCSP servers, and none of them is with multiple OCSP DPs. Details are shown in Table III.

Manual or out-of-band operations have to be involved in the revocation checks of locally-trusted certificates. 78 server certificates in the 118 locally-trusted certificate chains are self-signed, and the revocation of such certificates is to remove it from the CTL. Table III shows that, 61% of non-self-signed locally-trusted certificates are not configured with any CRI DP. Moreover, most CRI DPs configured in the other 39% are invalid (see Section II-C1 for details).

1) Accessibility of CRI:

a) Publicly-trusted certificates: 292 CRL DPs and 3,175 OCSP DPs are extracted from publicly-trusted certificates. Among these CRL DPs, 290 are accessible through normal networks, one is only accessible through encryption proxies, and the last is still unaccessible even through proxies. As for the 3,175 OCSP DPs, the numbers change to 3,097 DPs of 101 OCSP servers accessible through normal networks, 64 DPs of 3 servers accessible through proxies, and 14 DPs of 2 servers unaccessible. Considering the government censorship in some countries, CRI DPs unaccessible through normal networks but accessible only through special VPNs or proxies, are not occasional cases.

The only publicly-trusted certificates with the unaccessible CRL DP is configured with two accessible alternative DPs. Three DPs are with the same CRL server. The downloaded CRL files from the alternative DPs indicate that this certificate was revoked.

b) Locally-trusted certificates: There are 13 CRL DPs and 12 OCSP DPs in the locally-trusted certificates. Among the CRL DPs, only 2 are accessible through normal networks and 11 are unaccessible even through encryption proxies. As for the 12 OCSP DPs, the numbers change to 8 and 4, respectively. Most locally-trusted certificates with accessible CRI DPs are signed by the CAs established by well-known enterprises such as HUAWEI, China Mobile, and WoSign.[3] These CAs are not trusted in popular OSes and browsers by default.

Among the 11 unaccessible CRL DPs, 7 DPs are with LDAP which is usually deployed in private networks [29]. The access to other 4 unaccessible CRL DPs and 4 unaccessible OCSP DPs, ends with connectivity errors.

A study on the locally-trusted certificates on the Internet-wide scans [5] also shows that more than 99% are not configured with any CRI DP: 99.2% of locally-trusted certificates (or called “invalid” certificates) are not with any CRL DP and 99.9% are not with any OCSP DP. Therefore, in order to check the revocation status of these locally-trusted certificates, out-of-band means with prior knowledge are necessary.

2) CRI Measurement: We measured the time cost of CRI access, the size of CRL files and OCSP responses, and the validity period of downloaded CRI. Because not all TLS clients access CRI services when validating a certificate and such data are not reflected in the collected packets, we additionally access each accessible CRI DP 120 times (10 times every hour, from 9 a.m. to 9 p.m.) after the collection.

a) CRI overhead: The measurement results are listed in Tables IV and V. The average message size of CRL is 80 KB, the median is 20 KB, and the maximum is 3,072 KB. The distribution is as follows: 26.5% are less than 1 KB, 57.3% are in the range between 1 KB and 80 KB, and 11.0% are between 80 KB and 200 KB. The numbers of OCSP are: 1.52 KB (average), 1.22 KB (median), and 1.98 KB (maximum), respectively. 40.6% of OCSP responses are between 0.47 KB and 1 KB, and the others are greater than 1 KB.

[3]WoSign root CAs were removed from publicly-trusted CTLs in 2016 or 2017 due to security incidents, but some certificates signed by these CAs are still trusted [26]–[28].

TABLE IV
THE COST OF CRL ACCESS

	N_n	N_e	T	T_{max}	S	S_{max}
Publicly-trusted	290	1	1,512	13,127	80	3,072
Locally-trusted	2	0	645	1,205		

N_n: the number of CRI DPs accessible through normal networks;
N_e: the number of CRI DPs accessible through encryption proxies;
T/T_{max}: the average/maximum time cost of CRI access, in ms;
S/S_{max}: the average/maximum size of CRI, in KB.

TABLE V
THE COST OF OCSP ACCESS

	N_n	N_e	T	T_{max}	S	S_{max}
Publicly-trusted	3,097	64	411	1,438	1.52	1.98
Locally-trusted	8	0	500	685		

For the time cost of CRL access, the average is 1,512 ms, the median is 813 ms, and the maximum is 13,127 ms. 28.1% of CRL access takes less than 300 ms, 23.4% more than 300 ms but less than 900 ms, and 21.5% in the range between 900 ms and 1200 ms. The numbers of OCSP access change to 411 ms (average), 348 ms (median), and 1,438 ms (maximum), respectively. 22.6% of OCSP access takes less than 300 ms, and 61.3% more than 300 ms but less than 600 ms.

The results are consistent with the performance evaluation of OCSP in 2012 [11], but a little different from some other studies. The time cost of 304 CRL DPs measured by Netcraft in April, 2018 is 206 ms [30] and the median of OCSP access for Alexa Top-1000 websites in 2016 is about 20 ms [31]. The difference comes from the speedup by content distribution networks (CDNs) [30], [31]. We accessed the 304 CRL DPs measured by Netcraft but from our department: the average is 857 ms, the median is 249 ms, and the maximum is 30,683 ms. 55.3% of CRL access takes less than 300 ms, and 21.0% takes between 300 ms and 900 ms. 112 IP addresses of CRL DPs belong to CDNs, but 94.1% are not in China. Then, we analyzed the IP addresses of 108 available OCSP servers in the collected dataset: 62 addresses belong to CDNs, but 74% are not in our country. When we measured the OCSP access for Alexa Top-1000 websites from our department, the average time cost is 683 ms and the median is 409 ms. In another word, the users in our data collection do not significantly benefit from the speedup by CDNs.

CRI access is inefficient, compared with TLS handshakes. In our dataset, the average time cost of 2,046,594 full TLS handshakes is 240 ms, from the `ClientHello` message to the last `Finished` message. Similarly, the median cost of TLS handshakes measured in [31] is 242 ms.

b) Validity period: The validity period of the CRI for server certificates is generally shorter than that for CAs. Table VI shows the CRI validity periods for different certificates. Among the 209 CRL files for TLS server certificates, 74% are valid for a period of 7 days, the average is 7.34 days, the minimum is only 3 hours, and the maximum is 93 days. Among the 84 CRL files for intermediate CAs, 93% are valid for a period of 7 days or greater, the average is 133 days, and the maximum is 365 days. Among the 3,054 OCSP responses for server certificates, 79% are valid for 4 days, and the average is 4.32 days. For the 115 OCSP responses for CAs, 71% are valid for 7 days, and the average is 7.64 days.

We choose 7 and 133 days as the typical validity periods of the CRL files for servers and immediate CAs, respectively. As for OCSP responses, the values are 4 and 7 days. These typical validity periods will be referenced in the next analysis.

3) CRI Requirement by Users: As mentioned above, there are 3,187 certificates configured with OCSP DPs, and each appears 331 times averagely. Thus, although an OCSP response is used to check the revocation status of only one certificate, most OCSP responses are needed by multiple users. 305 unique CRL DPs are extracted from all 2,795 certificates, and a CRL DP includes the revocation status of several certificates. So each CRL file is needed by more users for more times than an OCSP response. Each of these CRL DPs appears 3,906 times averagely in the collection.

We analyzed the CRI requirements by different users within the validity period of CRI. The typical validity periods for different CRI are 4, 7, and 133 days, respectively. We divided the collection of TLS handshakes in 33 days into three non-overlapped subsets, according to these validity periods. For each subset, i.e., within a validity period of a CRL file or an OCSP response, we analyzed the requirement of each CRL file (or OCSP response), i.e., counted (*a*) the number of users that need it for revocation checks, and (*b*) the times that it is needed by these users. Then, we cumulated the numbers and times of all subset in every set. These results reflect the CRI requirements if revocation checks are fully implemented in TLS clients, but not the actual CRI access in the collected TLS sessions.[4] This analysis covers all CRI DPs, either accessible or unaccessible, and either preferred or alternative.

Figure 1 shows that a CRL file is needed for more times by more users than an OCSP response, and a CRL file (or OCSP response) for intermediate CAs is needed for more times by more users than that for servers. In Figures 1(a) and 1(b), 23% of the OCSP responses for servers are needed by multiple users and 40% are done for more than one times within the validity period of 4 days. As for the OCSP responses for intermediate CAs, the numbers increase to 61% and 74%, respectively.

Within the validity period of 7 days, 40% of CRL files for TLS servers are needed by multiple users and 59% are done for more than one times in Figure 1. For intermediate CAs, the numbers increase to 94% and 95%, respectively, i.e., almost every CRL file is needed for several times by multiple users.

D. The Results of Other Data Collections

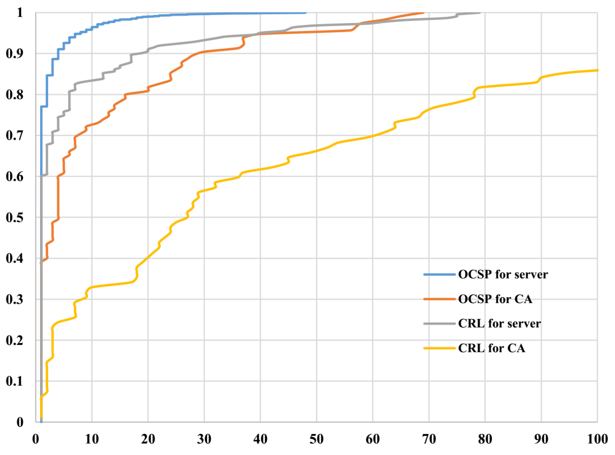
The locally-centralized design is proposed based on these analysis results: (*a*) only a limited number of publicly-trusted root CAs are involved in TLS sessions of a certain organization, so that the centralized CTL management removes most irrelevant publicly-trusted CA certificates to reduce the attack surface; (*b*) almost every locally-trusted root CA signs only

[4]This analysis should be performed, based on the validity period of each CRL file and OCSP response downloaded as the TLS handshake packets were being collected. However, not all CRI DPs were automatically accessed by TLS clients and we did not dump such data. We accessed the CRI services, after the data have been collected and when we were analyzing the certificates.

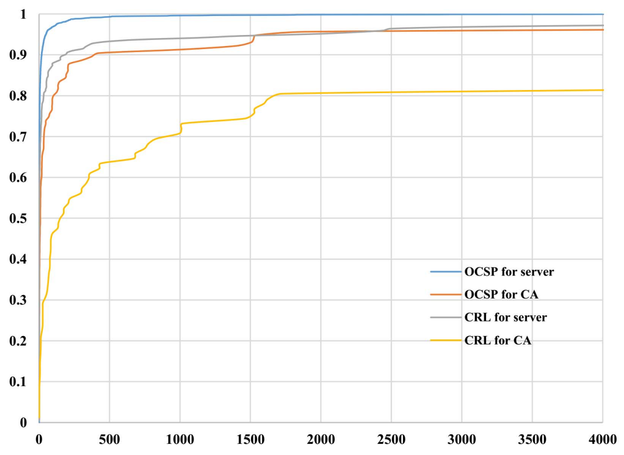
TABLE VI
ANALYSIS RESULTS OF CRI VALIDITY PERIODS

		N_{cri}	$V_{average}$	V_{min}	V_{max}	V_t	$P_=$	$P_<$	$P_>$
CRL	server	209	7.34	3 hours	93	7	74%	17%	9%
	CA	84	133.26	4	365	133‡	-	-	-
OCSP	server	3,054	4.32	15 mins	20	4	79%	8%	13%
	CA	115	7.64	4	20	7	71%	20%	9%

N_{cri}: the number of accessible CRI DPs extracted from certificates;
$V_{average}$/V_{min}/V_{max}: the average/minimum/maximum validity period of CRI, in days;
V_t: the typical validity period, selected with the largest proportion;
$P_=$/$P_<$/$P_>$: the proportion of CRI of which the validity period is equal/less/greater to/than V_t.
‡The CRL validity period for CAs varies greatly. 64% are greater than 33 days, 24% are between 100 and 166 days, and 19% are 365 days.



(a) the number of users that need each CRI



(b) the times that each CRI is needed by users

Fig. 1. The requirements of different types of CRI within the validity period (CDF).

one certificate chain, and then the locally-trusted certificates specified by each certificate chain in the CTL do not cause more efforts to the IT administrators; and (*c*) the CRI requirements of different users are similar, so the CRI caching among users greatly improves the performance of revocation checks.

We study other data collections to find whether these analysis results still hold. First of all, by comparing the 12 trust stores of all major platforms and 48 million certificates scanned using ZMAP [32], the study [2] shows that 34% of root CAs in the public CTLs do not sign any TLS certificate, and may be removed without hampering daily Internet activities. In the dataset of over 80 million certificates collected from 222 full IPv4 scans over three years [5], 70,637,981 TLS certificates are locally-trusted (or called "invalid" certificates). Among these invalid TLS certificates, 88.0% are self-signed root-CA certificates and 11.99% are signed by these root CAs. They are signed by 1.7 million unique keys, so on average every locally-trusted root CA signs only (70637981*11.99%)/1,700,000 = 4.98 certificate chains.

The 30-minute collection at the University of Auckland, New Zealand showed only 15 root CAs with 166 server certificates (i.e., 11 certificates per root CA) in TLS handshakes [33], which is roughly consistent with our result of 150 root CAs with 3,198 certificates (i.e., 21 certificates per root CA). In this dataset, each certificate chain averagely appears 13.2 times within half an hour [33]. No more details are disclosed, but it shall be done in working hours. It is reasonable to infer that most certificates were sent (26.4 times per hour) to different users, and then there is very remarkable similarity of CRI requirements of these users. Our data collection lasted for about 190 working hours, so each certificate chain averagely appears only 548,183/3,198/190 = 0.9 time per hour. Since our analysis shows that the CRI requirements of different users overlap, the overlap in the University of Auckland shall be much more remarkable (26.4 vs. 0.9 time per hour).

A. Bates *et al.* collected a full hour of university TLS traffic to evaluate Convergence notary services [34]. In this dataset, 77% of certificate validation triggers the client-side cache mechanism, 21% hits the notary-side caching, and only 1% results in cache misses in the notary service. This implies that, about 21/(21+1) = 95% of server certificates appear in TLS sessions of multiple users. Therefore, the CRI requirements of different users overlap remarkably. According to the TLS traffic from April 17 to June 17, 2018 of 1,000 individual users in Brigham Young University [35], 64.97% of certificates are needed for multiple times within one day by some single user, but the number increases to over 99% when the requirements by different users are compared (i.e., 99% of certificates are needed for multiple times within one day, by some single user or multiple users). So the similar CRI requirements also exist in this dataset.

Another dataset was collected at an institute with about 250 users [36]. There are 84 publicly-trusted root CAs in the total 36,949,381,778 TLS sessions during 11,383 hours (i.e., about 2,710 working hours). It also confirms our result that, only a small number of publicly-trusted root CAs are involved in the daily works. The dataset includes 46,560 certificates and 203,095,274 TLS sessions. Each certificate appears 203,095,274/46,560/2,710 = 1.6 times per hour, in working hours. It is roughly consistent with our result of 0.9 time per hour. It also implies the CRI requirements shall be probably consistent with our result, for the numbers of users in the datasets (250 vs. 218 users) are approximately equal.

On the other hand, the monitoring of three universities and affiliated research institutions [37] implies the very limited overlap of the CRI requirements in two weeks (about 560 working hours). That is, in the dataset from different organizations, the CRI requirements do not overlap so remarkably as that within one organization. In particular, 102,329 certificates appear 989,040 times (i.e., only 0.017 time per hour), or 77,361 ones appear 932,467 times (i.e., only 0.022 time) if Grid certificates are excluded. Note that two passive data sources are presented but there are lots of packet loss in the first one [37], so we only cite the second.

In summary, even on the scale of Internet, (*a*) the number of publicly-trusted root CAs involved in TLS sessions is very limited, compared with the hundreds of root-CA certificates installed in OSes or browsers by default, and (*b*) on average every locally-trusted root CA signs only one or two certificate chains. So the centralized CTL management with different levels of granularity for publicly-trusted and locally-trusted CAs will be suitable for more organizations. Meanwhile, the overlap of CRI requirements across different users generally holds within an organization, especially for an institute or university at least.

Therefore, existing studies imply and our data analysis shows that, due to the similarity of requirements among different users, the centralized management of PKI resources proposed in our scheme is applicable to the deployment in an organization to provide unified centralized management and resource sharing services. Even in the most extreme scenarios where each locally-trusted certificate is used only by one user, vCertGuard also provides a means to centrally manage CA certificates for users in the organization. This reduces the users' additional security configuration operations, but does not cause more efforts of the IT administrators.

III. Locally-Centralized Certificate Validation in Desktop Virtualization Systems

This section presents the principles of locally-centralized certificate validation, followed by the design and implementation of vCertGuard in desktop virtualization systems. We also discuss the possibilities to implement the locally-centralized CTL management and CRI access with other existing enhanced certificate validation schemes.

A. Design Principle

Our design goal is to facilitate the validation of TLS server certificates, especially for the end users within an organization, in a locally-centralized way. The principles are as follows.

1) Provide Secure and Flexible CTL Management Services: The CTLs of all users within an organization are configured by professional IT administrators. Only the IT administrators that understand the CTL requirements, are authorized to configure the CTLs, and users access the CTLs through read-only interfaces. This is more reliable and secure than manually managing CTL by non-professional end users. Meanwhile, with vCertGuard, IT administrators can configure specific CTLs for specific users that meet their customized needs. Moreover, trusted certificates are managed at different levels of granularity: the self-signed certificates of publicly-trusted CAs are listed, while locally-trusted certificates are specified by each certificate chain. This asymmetric design mitigates the potential threat of fraudulent certificates [2], [3], but does not cause more efforts of the IT administrators. This is another means of privilege restriction for CAs and IT administrators.

2) Improve the Accessibility of CRI and the Performance of CRI Access, and Mitigate the Privacy Leakage in CRI Access: The CRI access is centralized, and then IT administrators may obtain CRI by alternative channels (even out-of-band means), which is originally unaccessible to users. The CRI access is coordinated to (*a*) enable the sharing of downloaded CRI among users, of which the CRI requirements overlap remarkably, and (*b*) mitigate the privacy leakage, for the CRI request is sent by a group of users.

The locally-centralized certificate validation can be implemented differently. Based on the network environment and user devices, there are multiple corresponding methods, including push-based mechanisms and network interception strategies, or even some combinations of them. In another word, the implementation of locally-centralized certificate validation offers extensibility and compatibility.

B. vCertGuard Functionality

As mentioned above, the locally-centralized design can be implemented in various ways within an organization. We adopt the desktop virtualization systems as the base of locally-centralized solutions, which provide VMs for end users with centralized policies and controls.

Figure 2 shows the structure of vCertGuard. The vCertGuard services are implemented as virtual devices, accessible to the VMs. The vCertGuard virtual device is activated by VMMs, and mounted by a VM when it is booting. The back-end driver in the VMM responds to the vCertGuard request from the VM. Necessary resources of the vCertGuard services (i.e., CTLs, CRL files and OCSP responses) are stored in some special domains of the storage system, unaccessible to any VM. These resources are managed and configured by special IT administrators on the manager node of the virtualization platform (but not the administrators of any desktop).

VMMs emulate isolated peripheral devices for each VM, and implement resource management and sharing. The isolation feature of virtualization prevents the VM from unauthorizedly accessing memory data of the VMM, so that the VMM is generally considered more trustworthy than a VM. Meanwhile, a number of schemes have been proposed to

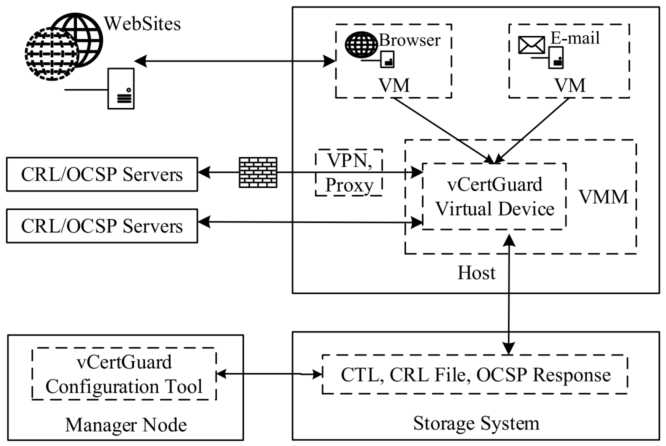


Fig. 2. vCertGuard overview.

implement VMM security enhancements, to protect against any unauthorized access to the VMM resources (i.e., to eliminate VM escape), which is initiated through the interfaces of VMM services. Such mechanisms include [38], [39]: (*a*) Securing the hypervisor (or VMM) from rootkits through integrity checking, such as HyperGuard [40], HyperCheck [41], HyperSentry [42], HyperSafe [43], and HyperVerify [44]; (*b*) Securing the hypervisor from rouge VMs, i.e., providing strong isolation to VMM resources from VMs or an isolated execution environment for a VM, including No-Hype [45], TrustOSV [46], Min-V [47], SplitVisor [48], Hyper-Lock [49], DeHype [50], and Nexen [51]; and (*c*) Securing the hypervisor from malicious VM users, i.e., securing the hypervisor from an untrusted launch of a VM image [52]–[54].

1) CTL Service: On receiving a request of the CTL service from VMs to the vCertGuard virtual device, the back-end module (*a*) answers whether the queried certificate is trusted, and signed by a publicly-trusted root CA or not if trusted, or (*b*) returns a self-signed certificate or a certificate chain in the CTL, if it exists. The request is sent along with the identifier of a certificate (e.g., the subject key identifier, or the issuer name with the certificate serial number).

2) CRI Service: On receiving a CRI DP from the VM to obtain a CRL file (or OCSP response), the back-end module checks the list of cached CRI and returns the requested CRI if it exists. If it does not exist, a *to-be-cached* entry is appended in the list and it attempts to access the CRI. After the CRI is downloaded and returned to the VM, the entry is marked with its expiration date; otherwise, if it fails even through all special network channels, this error has to be recorded in the log to wait for *manual* operations by IT administrators.

3) Configuration Tool: The CTLs, CRL files and OCSP responses are managed by IT administrators through the vCertGuard configuration tool. The tool is enabled, only after an IT administrator is authenticated with privileges in the desktop virtualization system. Firstly, the trust list of root-CA certificates (for publicly-trust certificates) and certificate chains (for locally trusted ones) are configured separately, i.e., add/remove a certificate or certificate chain to/from the CTL, which are simply stored in the storage system.

For the CRI service, vCertGuard maintains the list of cached CRI, and each entry consists of (*a*) its CRI DP, (*b*) the staus, (*c*) the downloaded CRI and the expiration date, and (*d*) network channels. The vCertGuard configuration tool periodically processes the list of cached CRI, and re-downloads the CRI that has expired or will expire soon. If a special channel (e.g., VPN or encryption proxy) is needed to access the CRI, also marked in the list, the tool will download it through this channel. Meanwhile, if an entry is not requested by any VM for several validity periods, it will be deleted from the list.

IT administrators check the log for unaccessible CRI DPs. They may mask unaccessible DPs by replacing them with the alternative but accessible DPs (see the accessibility analysis of CRL DPs in Section II-C1, as some of certificate configured with multiple alternative CRI DPs). Sometimes, the IT administrators have to manually obtain CRI by out-of-band means, and then put it into the system. For a locally-trusted certificate in the CTL but not configured with any CRI DP, the IT administrators may directly configure its revocation status in the list. Thus, when a user requests the revocation status of such a certificate, the back-end driver will return the result.

C. vCertGuard Implementation

We extend oVirt [16] to implement the vCertGuard configuration tool, and the vCertGuard virtual devices are implemented based on virtio [55] in KVM-QEMU [17], [18]. oVirt is a widely-used open-source desktop virtualization system. virtio is the de-facto standard for virtual devices in KVM-QEMU, to establish channels between a VM and the VMM.

1) vCertGuard Configuration Tool: oVirt engine runs on the management node, and provides interfaces for IT administrators to manage resources of the platform. Commands issued from the management node, are accepted by virtual desktop server manager (VDSM) on each host to configure hosts, networks and the storage system, and to execute VM life-cycle operations.

The vCertGuard configuration functions are implemented in oVirt. In particular, we extend oVirt engine to management vCertGuard resources in the storage system. The vCertGuard parameters and resources are input manually by administrators, and then transported from oVirt user interface (UI), oVirt engine, and VDSM, finally to the storage system. oVirt engine sends the message to VDSM in a XML-RPC request. On receiving the vCertGuard XML-RPC invocation from oVirt engine, VDSM generates the corresponding NFS command that will be accepted by storage systems. We extend oVirt CLI to execute scripts periodically to process the CRI resources (e.g., CRI update and delete).

2) vCertGuard Virtual Device: virtio defines a two-layer structure to implement virtual devices. virtqueue is a virtual message queue that conceptually connects a front-end driver that runs in VMs to the corresponding back-end driver in KVM-QEMU. Then, the frond-end driver and the back-end work cooperatively to emulate I/O devices.

The vCertGuard services run as virtual PCI devices in the VMs, using the virtio-over-PCI model. When the VM mounts a vCertGuard virtual device, a default channel (or pair of virtqueues) is assigned to initiate the virtual device, and then two service channels are established: one for the CTL service, and the other is for the CRI service. The front-end driver sends a request through virtqueues, and waits for the result from the back-end driver. The back-end driver is allowed to access the vCertGuard resources (i.e., CTLs, CRL files and OCSP responses in the system) with privileges.

A user-space program in the VM invokes the vCertGuard services by sending requests to the virtual devices, through the `ioctl` system call. We further encapsulate the `ioctl`-based commands into C-language APIs, to facilitate the invocation in browsers and other TLS clients. In the future, we will provide more friendly APIs, compatible with CryptoAPI and other certificate validation engines, to facilitate the deployment.

PKI clients, especially legacy non-open-source PKI application systems, may invoke the enhanced certificate validation functions in a variety of ways, including dynamically-loaded security libraries [56], [57], network interception modules [33], [35], [58], [59], GOT hooks [60], etc. These mechanisms can also be adopted to invoke the vCertGuard services. For instance, certificates that need to be validated can be obtained through the network interception modules, which then call the vCertGuard services to complete the certificate validation and returns the results to the TLS application. That is, there are flexible ways to integrate the vCertGuard services into PKI applications.

3) vCertGuard Storage: The vCertGuard resources are stored in the storage system as well as other resources in the desktop virtualization system, with the network file system (NFS) services. The vCertGuard resources are shared among hosts but not accessed directly by any VM (i.e., in the isolated virtual network of hosts). Then, the vCertGuard virtual devices in the VMM can access these resources by the file path that mounts the resource directory of the storage system.

D. Applicability of the Locally-Centralized Design

vCertGuard demonstrates the practicability of the locally-centralized design in desktop virtualization systems, but the design can be implemented with various system architectures, based on the network environment (e.g., high-speed LANs or WIFI) and user devices (e.g., desktop, desktop virtualization, laptop or mobile device). We discuss other application scenarios as follows.

It can be implemented as a network service accessed by the users within the organization, which works similarly to server-based certificate validation [61]. Then, the service manages the CTLs and caches the CRI for all users in a local area network (LAN), and a user queries the trustworthiness of a certificate or requests the CRI by a CRI DP. The LAN service provides an option for TLS clients to validate certificates. Alternatively, this design can be enforced as mandatory policies, for example, by patching the certificate validation procedure in legacy dynamically-linked applications [56], intercepting all TLS traffic in the OS kernel [58] or at the network gateway [33], to validate certificates based on the managed CTLs and cached CRI. The design can also be integrated into existing centralized security management solutions. For example, Windows AD Group Policy [62] manages CTLs for a group of desktops in a centralized way. Although it does not support the asymmetric management of publicly-trusted and locally-trusted certificates or the CRI caching among users, our design principles are compatible with the service mode and can be implemented in Windows Group Policy.

Among the above approaches which implement enhanced certificate validation policies, the TLS interceptor at the network gateway [33] shall be the most applicable to different systems. Other approaches work with some limitations. For example, CertShim [56] is effective only for dynamically-linked programs, Windows AD Group Policy [62] only works for Windows, TrustBase [58] does not provide methods to share CRI among different computers, and server-based certificate validation [61] cannot enforce mandatory policies. Although our policies of locally-centralized CTL management and CRI caching shall be applicable to different networks and systems including WIFI and mobile devices, vCertGuard only works in desktop virtualization systems which are usually deployed on high-speed LANs. On the other hand, in wireless scenarios, the CRI caching which saves the network traffic greatly to access CRI services, will be more attractive. Push-based mechanisms (i.e., CRLite [10], CRT [35], and Let's Revoke [63]) and network interception strategies (i.e., RITM [59] and CRG [33]) can be applied to mobile devices, to implement the locally-centralized design. The former needs to explicitly modify PKI client software in the mobile devices, but can work in any network. The latter does not require any modifications of PKI clients; however, due to the nature of using middle-box, the mobile devices lose protections when they leave the network [63].

vCertGuard, Windows AD and other server-based solutions enable professional IT administrators to manage PKI resources centrally and efficiently. Only special administrators are authorized to configure these resources; i.e, the administrators of hosts but not VMs, of Windows AD but not any Windows desktops, of the validation server but not clients. After configuring the CTL, AD Group Policy pushes the list to each Windows desktop within the domain, and the CTL is used by Windows CryptoAPI. Meanwhile, vCertGuard completes the CTL configuration in the VMM, and then a VM requests the vCertGuard CTL service through the encapsulated API.

Compared with AD Group Policy and other online server-based approaches, vCertGuard brings the following advantages: (*a*) no additional hardware resources are required (e.g., TLS interceptor or AD domain server), and then no extra device with potential attack surfaces is introduced; (*b*) it is more efficient without connections to an extra network device; and (*c*) it prevents privacy leakage to OCSP servers, when requesting OCSP responses. Note that, when vCertGuard is deployed in the VMM of desktop virtualization systems, the VMM inherently has the privileges to monitor all the network traffic of the VMs and knows all privacy behaviors of the VMs. So vGertGuard does not introduce extra privacy leakage risks.

vCertGuard, as a locally-centralized solution, currently only works in desktop virtualization systems, and it requires certain modifications of PKI clients to invoke the vCertGuard services. Provided that vCertGuard runs in a virtualized environment of an organization, it is reasonable to assume that IT administrators can modify the whole software stack of VMs to deploy vCertGuard, avoiding the software modifications done by each end user. As mentioned above, in addition to the invocation of the `ioctl`-based APIs, the adoption of LD_PRELOAD, GOT hooks, and network interception modules helps to integrate vCertGuard services into non-open-source PKI applications. To simplify this process, vCertGuard's API could be invoked by existing solutions that intercept certificate validation. For instance, a plugin of CertShim or TrustBase can query the vCertGuard API for PKI resources (i.e., CTL and/or CRI), and deny or allow the TLS connection based on the response.

In summary, vCertGuard demonstrates a different way to enforce centralized certificate validation policies, compared with exiting approaches [33], [56], [58], [61], [62].

IV. Evaluation

The evaluated prototype system consists of: (*a*) a manager node with CentOS v7.0 as the OS; (*b*) two Dell PowerEdge R730 servers as the hosts, each of which is configured with an Intel Xeon E5-2609 v3 1.90GHz CPU and 16GB RAM; and (*c*) an NFS server (IBM System x3550 M4) as the storage system. We installed oVirt v4.1, KVM v0.12.1 and QEMU v1.7.1. One VM is created and assigned with 4 vCPUs and 8GB RAM, and its OS is CentOS v7.0. The hosts and the VM are connected to the 100Mbps Ethernet of our department, and share the 1Gbps port to the Internet with other desktops.

A. Invoking the vCertGuard Services

The vCertGuard services are evaluated by Chromium in the VM. On Linux, Chromium uses the Mozilla NSS library as its certificate validation engine, which implements the functions of certificate path, CTL management, CRI access and revocation check. We modified Mozilla NSS v3.26.2 in Chromium 54.0.2830.0, to invoke the vCertGuard services when it validates a TLS server certificate.

1) CTL Service: According to the certificate chain built by Mozilla NSS, Chromium invokes the vCertGuard CTL service in different ways. If a complete chain is built for the received certificate with a root CA in the CTL of Mozilla NSS, Chromium invokes the CTL service by sending (the identifier of) the root-CA certificate, to check whether the CA is trusted in vCertGuard as a publicly-trusted root CA or not. Otherwise, if it cannot build a complete certificate chain with the CTL of Mozilla NSS, Chromium invokes the CTL service by sending the identifier of the server certificate, attempting to return the locally-trusted certificate chain from the vCertGuard CTL.

2) CRI Service: Mozilla NSS implements its own CRI cache mechanism: the CRI is cached in memory for future revocation checks, until the CRI expires or Chromium is closed. Mozilla NSS checks the revocation status as follows. It tries to find the CRI in its cache according to the CRI DP; then, if it is not cached, it accesses the CRI server to obtain the OCSP response (or CRL file). Whether the CRI is cached or not, it always prefers OCSP responses to CRL files.

TABLE VII
The Time Cost of the CTL Service (in μs)

		# of certificate (chain)		
	In the CTL	10	150	1000
`CTL-Serv1`	Yes	57	97	173
	No	66	113	211
`CTL-Serv2`	Yes	233	315	420
	No	63	107	234

We extended the CRI access in Mozilla NSS as follows. If a CRI DP is configured in the certificate and the unexpired CRI is not cached, Mozilla NSS invokes the vCertGuard service to obtain the CRI, instead of through networks by itself.

In the evaluation experiments, the browser implements *full revocation checks for each certificate*, different from some browsers that do not support CRL and/or OCSP for some certificates. The main reason of disabling revocation checks in browsers is the inefficiency of CRI access [10], [13], [14]. So we investigate the overheads of revocation checks with vCertGuard in the "perfect" scenario of full revocation checks to comprehensively learn the performance improvements, and then, if the measured overheads are acceptable in this extreme scenario, revocation checks become more feasible with the improvement of CRI caching within an organization. Finally, we evaluate vCertGuard only with browsers, because, as the typical TLS client, a browser also implements some CRI cache mechanism by itself, so that we can analyze more complicated scenarios (i.e., the scenarios of Autocached and Uncached in the next).

B. Evaluation of CTL Service

Two functions of the vCertGuard CTL service are evaluated: (*a*) `CTL-Serv1` checks whether a CA certificate is trusted as a publicly-trusted root CA or not, and (*b*) `CTL-Serv2` returns a certificate chain from the CTL. In the evaluation, different numbers of certificates (or certificate chains) are set in the CTL. Chromium queried a certificate (or certificate chain) randomly listed in the CTL or not, and we measured the extra time overhead of the vCertGuard service (starts when it sends the request and ends when the result returns). We evaluated each case for 1,000 times.

Table VII lists the average results. The overhead of the CTL service is rather negligible, compared with TLS handshakes. The time cost of `CTL-Serv1` is not greater than 211 μs and the cost of `CTL-Serv2` is always less than 420 μs, even when there are 1,000 certificate chains in the CTL. Note that, the average time of a full TLS handshake is about 240 ms.

C. Evaluation of CRI Service

The evaluation is performed on 293 CRL DPs and 3,169 OCSP DPs accessible through normal networks or encryption proxies, based on the CRI requirements of collected data.

Three cases are compared: (*a*) CRI access through network channels directly, (*b*) through the vCertGuard CRI service when the requested CRI is not cached, and (*c*) through the vCertGuard service when cached. The details of direct CRI

TABLE VIII
THE TIME COST (IN ms) OF CRI ACCESS IN CHROMIUM

	Through network channels directly			Through vCertGuard					
				CRI is not cached			CRI is cached		
	Average	Interval	Median	Average	Interval	Median	Average	Interval	Median
CRL	1,512	124 - 13,127	813	1,543	167 - 13,203	833	11	7 - 45	15
OCSP	411	37 - 1,438	348	423	56 - 1,476	361	4	1 - 17	10

access through network channels are included in Section II-C2 (Tables IV and V). Similarly, the data of CRI access through the vCertGuard service (when it is not cached), are measured 120 times (10 times every hour, from 9 a.m. to 9 p.m.).

In Table VIII, the extra overhead is negligible when the requested CRI is not cached, because the time increases averagely by only 2.1% and 2.9% for CRL DPs and OCSP DPs, respectively. When the CRI has been cached in vCertGuard, the time to obtain CRI is reduced by 99.3% for CRL DPs and 99.0% for OCSP DPs, compared with the access directly through network channels. When CRI is cached, it is more efficient to access an OCSP response than a CRL file (4 vs. 11 ms) through the vCertGuard service due to the size.

To invoke the vCertGuard CRI service, the time cost mainly results from: (*a*) the communication between the host and the storage system to return the cached CRI resources; (*b*) the process by the back-end driver in the VMM; and (*c*) the process by the front-end module and the `ioctl` system call in the VM. The above cost becomes greater as the message size increases, and the average size of CRL files is more than 50 times that of OCSP responses (see Section II-C2). When CRI is not cached, the overhead is mainly caused by the inefficient access to the CRI servers through network channels, as shown in Table IV. One of our future work is to optimize the implementation; e.g., prefetch CRL files and OCSP responses.

Next, we analyze the performance improvement magnified due to the similarity of CRI requirements of different users. Section II-C3 shows that, either for CA certificates or server certificates, a CRL file (or OCSP response) is needed by multiple users during its validity period. Although the CRI cache mechanism is implemented in browsers and OSes [11], [29], it does not share CRI among different desktops. Of some browsers (e.g., Internet Explorer, Safari, Chrome in Windows and Mac OS X), the cached CRI is maintained by OSes and cleaned on expiration; while others (e.g., Firefox on all platforms, Chrome on Linux) cache CRI in memory, and the cached CRI is cleaned when the browsers are closed.

We evaluate vCertGuard in two extreme scenarios: (*a*) once downloaded, the CRI is cached on the desktop until it expires, denoted as *Autocached*, and (*b*) a browser accesses the CRI service every time it needs the CRI, denoted as *Uncached*, i.e., CRI is never cached on browsers or VMs. The CRI caching in practice is a hybrid of these extreme scenarios. To fully evaluate the CRI services, we assume a browser that strictly checks the revocation status of each certificate with CRI DPs. The access of CRL and OCSP is evaluated separately; that is, we ignored all OCSP DPs when analyzing the CRL access, and ignored CRL DPs when the OCSP access is analyzed.

TABLE IX
THE REQUIREMENTS OF ACCESSIBLE CRI DPS

		N_{cri}	N_r	N_u
CRL	server	209	622,172	7,080
	CA	65	668,658	2,985
OCSP	server	3,054	523,617	36,687
	CA	115	543,753	7,179

N_{cri}: the number of unique CRI DPs, each of which appears as the first accessible CRI DP in a certificate;
N_r: the number of CRI requirements, cumulated by each validity period;
N_u: the number of users that need the CRI, cumulated by each validity period.

We applied the collected TLS sessions to these scenarios, with and without vCertGuard, and then calculated the overhead to obtain the needed CRL files and OCSP responses in their validity periods, through the first accessible CRL and OCSP DPs of each non-self-signed certificate. The collection of TLS handshakes in 33 days was divided into three sets of non-overlapped subsets, according to different typical CRI validity periods (4, 7, and 133 days; see Section II-C2). For each subset of the data in a set (i.e., within the validity period of a CRL file or an OCSP response), we analyzed the cumulated overhead of each CRI access, according to (*a*) the size and download time of this CRI, (*b*) the requirements of this CRI by the users that need it for revocation checks, and (*c*) the average extra overhead of CRI access through vCertGuard (i.e., 31/11 ms for CRL access and 12/4 ms for OCSP access when uncached/cached; see Table VIII).

There are totally 305 CRL DPs and 3,187 OCSP DPs, while 12 CRL DPs and 18 OCSP DPs are unaccessible (see Sections II-C). 20 accessible CRL DPs appear in CA certificates as alternative DPs. We keep the only alternative CRL DP that appears in the certificate with an unaccessible preferred CRL DP. Finally, we remove 19 alternative CRL DPs and all unaccessible DPs, and so only 209/65 CRL DPs and 3,054/115 OCSP DPs for servers/CAs are evaluated, as a browser only needs the first accessible DP for revocation checks. The removal of unaccessible and alternative CRI DPs makes the evaluation be closer to real-world scenarios, and the removed DPs compose only a small proportion. Table IX shows the accessible CRI requirements by different users, cumulated by each validity period of the 33 days.

The cumulated overheads of time cost and network traffic for each case, are shown in Tables X and XI. Either for CRL files or OCSP responses, whether a browser caches CRI or not, vCertGuard significantly improves the performance of CRI access by one, two or even more orders of magnitude, in terms of time cost and network traffic.

TABLE X
CUMULATED TIME COST (IN SECOND) OF THE CRI DPS

	w/o vCertGuard		w/ vCertGuard			
			Autocached		Uncached	
	Autocached	Uncached	Total time	Invocation time	Total time	Invocation time
CRL for server	7,850.412	804,686.604	803.828	110.275	7,569.848	6,876.287
CRL for CA	983.072	103,570.591	69.806	34.850	7,392.206	7,357.253
OCSP for server	18,644.199	261,696.268	5,277.319	439.932	7,231.036	2,387.652
OCSP for CA	3,517.720	246,555.173	262.765	35.616	2,409.061	2,181.912

TABLE XI
CUMULATED NETWORK TRAFFIC (IN MB) OF THE CRI DPS

		w/o vCertGuard		w/ vCertGuard
		Autocached	Uncached	
CRL	server	2,068.595	178,156.766	85.071
	CA	2.440	566.787	0.047
OCSP	server	40.894	564.612	10.534
	CA	8.101	640.555	0.491

The improvement on network traffic is more significant than that on time cost, mainly because there is extra invocation time introduced by vCertGuard but no extra "invocation network traffic". In the Autocached scenario, the network traffic to access CRL DPs (and OCSP DPs) for servers is reduced to only 4.1% (and 25.8%, respectively), while the time cost is reduced to 10.2% (and 28.3%), where the invocation accounts for 1.4% (and 2.4%, respectively). In the Uncached scenario, the invocation becomes the main cost, because the browser has to invoke vCertGuard every time it validates certificates. The invocation accounts for 99.5% to obtain the CRL files and 90.6% to obtain the OCSP responses for intermediate CAs.

Generally, if the CRI is needed by more users for more times and the validity period is longer, the improvement by vCertGuard becomes greater. The improvement of CRL access is better than that of OCSP access. In the Autocached scenario, the total time and traffic are reduced by 90.1% (from 8,833.484 seconds to 873.634) and 95.9% (from 2,071.035 MB to 85.118) for all CRL DPs, respectively, and the numbers for OCSP DPs are 75.0% (from 22,161.919 seconds to 5,540.084) and 77.5% (from 48.995 MB to 11.025). In the Uncached scenario, the improvement in each case increases to 98.1% or higher, and the improvement of CRL access is also greater than that of OCSP. The greatest validity period is 133 days of the CRL files for CAs. Thus, in either of the two scenarios, the total network traffic of vCertGuard to obtain the CRL files for CAs is the least, and the improvement is the greatest. This conclusion also holds for the time cost in the Autocached scenario. Meanwhile, in the Uncached scenario, the time cost with vCertGuard is almost equal to the invocation time in all cases, except the OCSP responses for servers. The validity period of the OCSP responses for servers is the least (i.e., 4 days), and then the time cost to update OCSP responses is non-negligible (compared with the invocation time).

The improvement by the CRI caching design of vCertGuard in the Uncached scenario, on either time cost or network traffic, is more significant than that in the Autocached scenario, because the CRI has been shared for more certificate validation by each user in the Autocached scenario. The only exception is the time improvement of the CRL DPs for CAs, where the invocation of vCertGuard services accounts for 99.5% of the time cost; if the invocation time is uncounted, this conclusion also holds for the CRL access for CAs.

Finally, when calculating the overhead of vCertGuard, we also counted the time of each CRI update triggered by the request from VMs. In the real-world deployment, the update of expired CRI will be performed automatically, so the time cost will be further reduced.

D. Extended Comparison

1) Comparison With CRLSet: We compare vCertGuard with CRLSet [64], a typical certificate revocation solution based on the push model. The CRLSet contains a selection list of revoked certificates, and the list is updated and pushed to browsers periodically. The CRLSet list is limited in size, and then it has low coverage [10], [29]. We took the CRLSet list on August 24, 2017 as a sample. It includes 15,172 revoked certificates signed by 59 CAs. These revocation entries also appear in 44 CRL files of the ones obtained from the 274 DPs in this evaluation. So CRLSet will be an alternative of only 44 CRL DPs in our dataset, and these CRL DPs are extracted from 166 certificates. Note that there are 3,246 non-self-signed certificates in the collected dataset.

Then, we applied CRLSet in the two extreme scenarios, and assumed that the browser always prefers CRLSet. The average time cost of performing revocation checks with CRLSet, is 183 μs. It is found that, in any case, the improvement by vCertGuard is always much greater than that by CRLSet. In the Uncached scenario, the total time and traffic are reduced by only 8% (from 908,257.197 seconds to 835,570.789) and only 0.9% (from 178,723.554 MB to 177,199.130) for CRL access, respectively. The numbers for OCSP access are 41.8% (from 508,251.441 seconds to 295,968.786) and 43.9% (from 1,205.167 MB to 676.672). In the Autocached scenario, the improvement in each case is always less than 13.9% for CRL access, and for OCSP access it is less than 10.8%. Note that, the cost of CRLSet access is not counted in this comparison.

2) Comparison With CDN Services: We briefly evaluate the vCertGuard CRI service, under an assumed scenario where the speedup of CDN servers works. Assume that, it uniformly takes 206 ms to download a CRL file, and 20 ms to access an OCSP response [30], [31] (see Section II-C1). Then, in the Autocached scenario, vCertGuard will save 85.5% (and 92.0%) of time to download the CRL files for servers (and for intermediate CAs, respectively). For the OCSP access,

TABLE XII
THE TIME COST OF CERTIFICATE VALIDATION IN CHROMIUM WITH AD GROUP POLICY (IN ms)

		Average	Interval	Median
Initialization	`CTLEng`	4.136	1.247 - 33.285	3.490
Validation	`C--`	2.877	0.547 - 21.937	2.102
	`CS-`	3.137	1.832 - 23.382	2.395
	`CSO`	837.111	2.477 - 15,509.732	533.329

it will save 54% and 73.6% for servers and intermediate CAs, respectively. In the Uncached scenario, vCertGuard will save 94.1% (and 94.2%) to access the CRL files for servers (and for CAs, respectively). For the OCSP access, it will save 78.2% (and 79.9%) for servers and for intermediate CAs, respectively.

3) Comparison With Windows AD Group Policy: We compare vCertGuard with Windows AD Group Policy [62], a typical CTL management solution based on the push model. AD manages CTLs for a group of Windows desktops in a centralized way, and the list is updated and pushed to the desktops periodically.

We modified Chromium in Windows, to evaluate the certificate (chain) validation services with AD Group Policy. AD Group Policy only affects the CTLs in Windows but not the certificate validation process. In Windows, Chromium uses system-provided CryptoAPI as its certificate validation engine, and CryptoAPI implements functions based on the CTL managed by AD. So we measured the certificate validation of Chromium to evaluate the effect of AD Group Policy. Four cases are measured: (*a*) `CTLEng`: create a certificate chain validation engine by calling `CertCreateCertificateChainEngine` of CryptoAPI, which is performed at least once during the initialization; (*b*) `C-`: build and validate the certificate chain by calling `CertGetCertificateChain` of CryptoAPI, where the revocation status is checked with the cached CRI of CryptoAPI; (*c*) `CS-`: in addition to (*b*), check the revocation status with the CRLSet list against the certificate chain, where the CRLSet lists are periodically retrieved and cached by Chromium; and (*d*) `CSO`: in addition to (*c*), the certificate chain validation allows online revocation checks by setting the flags to `CertGetCertificateChain`.

The measurements in Table XII are performed on Alexa Top-1000 websites accessed from our department. The results reflect the time cost of certificate validation in Chromium with AD Group Policy, and the operation of certificate validation takes about 3 ms, without about 4 ms for the initialization. Compared with the results in Table VII, the overhead of vCertGuard CTL services is negligible (less than 420 μs). It is seen in Table XII, with online revocation checks, the overhead increases significantly (i.e., the average time cost is 837 ms). So vCertGuard works cooperatively in the first two cases with negligible overheads, and improves the performance remarkably in the third case (in Table VIII, when the CRI has been cached in vCertGuard, the time to obtain CRI is reduced by more than 99%).

Summary: Compared with the overhead of the entire TLS handshake process (i.e., about 240 ms), vCertGuard only introduces a negligible additional overhead to invoke the CTL service (less than 420 μs), and provides an efficient and convenient approach of CTL management. Compared with accessing CRI through network channels directly, invoking the vCertGuard CRI service also introduces only a negligible overhead when the requested CRI is not cached. When the CRI has been cached in vCertGuard, the time to obtain CRI is reduced by more than 99.0%.

The performance improvements introduced by the vCertGuard are further amplified due to the similarity of the CRI requirements of different users. Whether it is a CRL file or an OCSP response, whether the browser caches CRI or not, vCertGuard improves the performance of CRI access in terms of time cost and network traffic. Generally, if the CRI is needed by more users for more times and the validity period is longer, the improvement of vCertGuard will become greater.

V. RELATED WORK

A. TLS Certificate on the Internet

Several studies were conducted on the TLS ecosystem. The active scans over the Internet [65] finish the certification structure of TLS certificates, while Holz *et al.* analyzed the certificate distribution among IP addresses and the quality of certificates [37]. The trust relationships among root CAs, intermediate CAs, and TLS servers are also studied [6]. Amann *et al.* [36] analyzed the benign changes of the trust relationships in the ecosystem, and improved the understanding of TLS man-in-the-middle (MitM) attacks. VanderSloot *et al.* [66] discussed the methodologies to finish a complete view of certificates on the Internet. We study TLS certificates on a different scale, and analyze the similarity of certificates validated by different users.

Among the "invalid" certificates on the Internet, 88.0% are self-signed locally-trusted certificates and 11.99% are certificates signed by locally-trusted root CAs [5]. Lots of organizations sign certificates for their own websites or internal services, as it is difficult or costly to deploy a "valid" certificate [25], [37]. These results are confirmed in our study.

B. CTL Management

34% of root CAs in the public CTLs do not sign HTTPS certificates, and may be removed without hampering daily Internet activities but reducing the attack surface [2]. It is confirmed in our study, as only a small number of publicly-trusted root CAs are in the collected TLS sessions.

Vallina-Rodriguez *et al.* [67] analyzed the root-CA certificates in the Android certificate stores, installed by manufacturers, mobile operators, and the Android OS. Root-CA certificates could be installed by attacks [68]–[70], or by parental-control and antivirus software [67], [71], [72]. Schemes have been proposed to restrict the number of trusted CAs based on the usage of CAs [2], or the trust view of other users [73]. vCertGuard proposes to manage the trust list in multiple levels of granularity: a certificate is trusted if it is signed by a publicly-trusted root CA in the CTL, while trusted locally-trusted certificates are specified by each certificate chain.

C. Certificate Revocation

Stark *et al.* [11] demonstrated that OCSP services are inefficient and then proposed to prefetch and prevalidate certificates to reduce the TLS handshake latency. Liu *et al.* [29] conducted a comprehensive performance study on revocation check services in 2015, and the access to OCSP servers is found to be improved by CDNs [31]. There are discussions about the performance of revocation checks in practice [10], [13], [14], because the CRI access is generally inefficient and causes privacy leakage.

Certificate revocation mechanisms based on the push model [10], [64], [74], can work cooperatively with vCertGuard to push revocation status to the back-end module. CRLite [10] aggregates CRI into a space-efficient data structure and proactively pushes it to browsers, so the performance improvement will be greater than CRLSet [64] and OneCRL [74]. RevCast [75] broadcasts revocation status through FM radio, while RITM [59] distributes CRI to the network middle-boxes which relay TLS handshakes and inserts CRI into handshake packets. Finally, CRG [33] implemented centralized and mandatory certificate revocation checks by intercepting TLS traffic.

Certificate revocation table (CRT) combines CRI caching and prefetching on the scale of organizations [35]. Each user maintains a table of the revocation status of certificates that are validated in recent TLS sessions, by periodically downloading CRL files and OCSP responses. The tables are shared with multiple users of an organization. Then, they collected the TLS traffic of Brigham Young University to simulate the inputs, to evaluate the performance of CRT. This work was published in October 2018, after we finished this manuscript. Our design is proposed based on the locality of both CTL management and CRI access, while CRT only considers revocation checks. More importantly, the CRI caching at the organization level is proposed as a deployment option in CRT [35], and we focus on the similar requirements of CTL management and CRI access by different users within an organization.

D. Certificate Validation

Schemes were proposed to tame the trustworthiness of CAs in certificate validation, based on the top-level domains certified [76], the countries of certificate subjects [77], the multipath probing of notaries [78], and the policies of certificate subjects [79]. Elaphurus [80] proposed to integrate multiple existing defenses in browsers, to detect fraudulent certificates in TLS handshakes.

Vulnerabilities of TLS certificate validation in libraries and applications have been discovered [57], [81], [82]. SSLint [83], SMV-Hunter [84] and MalloDroid [81] detect the certificate validation vulnerabilities on Linux and Android. CertShim [56] performs binary instrumentation to patch the vulnerabilities in legacy systems. Frankencerts [85] generates synthetic certificates for the large-scale testing of the certificate validation logic. TrustBase [58] implements a TLS traffic interceptor, enforcing mandatory practices of certificate validation. A. Bates *et al.* [34] evaluated the performance of Convergence notary services by mapping a full hour of university SSL traffic into the Convergence workload.

E. TLS and HTTPS Deployment

User studies were conducted with IT administrators to analyze the usability of TLS and HTTPS, and found that the deployment is complex even for professionals [25], [86]. In 2011, only 5.7% of the Alexa Top-1M websites correctly deployed HTTPS [87]. The TLS deployments for email and chat are shown to be insecure with weaknesses in the cryptographic setup [88].

VI. Conclusion

We finished a study on 218 users of an organization to analyze the requirements of CTL management and CRI access in the TLS sessions of daily works. The study shows that, only a small number of publicly-trusted root CAs but a great number of locally-trusted root CAs are involved in these TLS sessions, and almost every locally-trusted root CA signs only one certificate chain. It also shows that, the CRI services of locally-trusted certificates are usually unavailable, and the access of available CRI services (for either publicly-trusted or locally-trusted certificates) is inefficient. Meanwhile, lots of certificates are validated by different users for several times, so a CRL file (or OCSP response) is likely to be needed by different users for multiple times during its validity period.

Based on the analysis, we propose vCertGuard, a locally-centralized solution to solve the problems of both CTL management and CRI access with desktop virtualization systems. The vCertGuard services are designed and implemented, utilizing (*a*) the centralized management feature of desktop virtualization systems and (*b*) the similar requirements of CTL management and CRI access of different users within an organization. vCertGuard provides interfaces for professional IT administrators to remove unnecessary publicly-trusted root CA certificates and add locally-trusted certificate chains, not disturbing the desktop users. vCertGuard accesses the CRI servers for all VMs, so that the downloaded CRL files and OCSP responses are cached and shared among all VMs. The CRI sharing among users within an organization, improves the performance of CRI access significantly and mitigates the privacy leakage in CRI access. Our evaluation shows that, compared with a "perfect" browsers that fully performs revocation checks, the vCertGuard CRI service reduces the time cost of OCSP and CRL access by 90.1 - 98.4% and 75.0 - 98.1%, respectively, and the network traffic by 99.1% - 99.5% and 77.5% - 99.1%, respectively.

Acknowledgments

The authors would like to thank the editor and anonymous reviewers for their helpful comments on the paper.

References

[1] D. Cooper *et al.*, *Internet X. 509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, document RFC 5280, 2008.

[2] H. Perl, S. Fahl, and M. Smith, "You won't be needing these any more: On removing unused certificates from trust stores," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2014, pp. 307–315.

[3] GlobalSign. (2011). *Security Incident Report*. [Online]. Available: https://www.globalsign.com/resources/globalsign-security-incident-report.pdf

[4] K. Wilson. (2015). *Distrusting New CNNIC Certificates*. [Online]. Available: https://blog.mozilla.org/security/2015/04/02/distrusting-new-cnnic-certificates/

[5] T. Chung *et al.*, "Measuring and applying invalid SSL certificates: The silent majority," in *Proc. ACM Internet Meas. Conf. (IMC)*, 2016, pp. 527–541.

[6] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS certificate ecosystem," in *Proc. Conf. Internet Meas. Conf. (IMC)*, 2013, pp. 291–304.

[7] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor, "Crying wolf: An empirical study of SSL warning effectiveness," in *Proc. 18th USENIX Secur.*, 2009, pp. 399–416.

[8] D. Akhawe and A. P. Felt, "Alice in warningland: A large-scale field study of browser security warning effectiveness," in *Proc. 22nd USENIX Secur.*, 2013, pp. 257–272.

[9] D. Eastlake, *Transport Layer Security (TLS) Extensions: Extension Definitions*, document RFC 6066, 2011.

[10] J. Larisch, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "CRLite: A scalable system for pushing all TLS revocations to all browsers," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 539–556.

[11] E. Stark, L. S. Huang, D. Israni, C. Jackson, and D. Boneh, "The case for prefetching and prevalidating TLS server certificates," in *Proc. 19th NDSS*, 2012, pp. 1–15.

[12] E. Topalovic, B. Saeta, L. S. Huang, C. Jackson, and D. Boneh, "Towards short-lived certificates," in *Proc. 6th Workshop W2SP*, 2012, pp. 1–9.

[13] A. Langley. (2014). *No, Don't Enable Revocation Checking*. [Online]. Available: https://www.imperialviolet.org/2014/04/19/revchecking.html

[14] A. Langley. (2014). *Revocation Still Doesn't Work*. [Online]. Available: https://www.imperialviolet.org/2014/04/29/revocationagain.html

[15] C. R. Taylor and C. A. Shue, "Validating security protocols with cloud-based middleboxes," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2016, pp. 261–269.

[16] oVirt. (2018). *A More Powerful and Flexible Open Source Virtualization Solution*. [Online]. Available: http://www.ovirt.org/

[17] The KVM Project. (2018). *KVM: Kernel-Based Virtual Machine*. [Online]. Available: http://www.linux-kvm.org/

[18] QEMU. (2018). *The FAST! Processor Emulator*. [Online]. Available: https://www.qemu.org/

[19] Censys. (2018). *The Security Platform Which Finds and Analyzes Every Device Connected to the Internet*. [Online]. Available: https://censys.io/

[20] Microsoft Corporation. (2017). *Microsoft Trusted Root Certificate Program: Participants*. [Online]. Available: https://gallery.technet.microsoft.com/Trusted-Root-Certificate-38d12b11

[21] Mozilla. (2018). *Mozilla Included CA Certificate List*. [Online]. Available: https://wiki.mozilla.org/CA/Included_Certificates

[22] Apple Inc. (2017). *Lists of Available Trusted Root Certificates in macOS*. [Online]. Available: https://support.apple.com/en-us/HT202858

[23] R. Sleevi, "Sustaining digital certificate security," Google Online Secur. Blog, Tech. Rep., 2015.

[24] A. Langley, "Further improving digital certificate security," Google Online Secur. Blog, Tech. Rep., 2013.

[25] S. Fahl, Y. Acar, H. Perl, and M. Smith, "Why eve and mallory (also) love webmasters: A study on the root causes of SSL misconfigurations," in *Proc. 9th ACM Symp. Inf., Comput. Commun. Secur. (ASIA CCS)*, 2014, pp. 507–512.

[26] Microsoft Corporation. (2017). *Microsoft to Remove WoSign and StartCom Certificates in Windows 10*. [Online]. Available: https://cloudblogs.microsoft.com/microsoftsecure/2017/08/08/microsoft-to-remove-wosign-and-startcom-certificates-in-windows-10/

[27] Apple Inc. (2017). *Blocking Trust for WoSign CA Free SSL Certificate G2*. [Online]. Available: https://support.apple.com/en-us/HT202858

[28] Mozilla. (2017). *Removing Disabled WoSign and StartCom Certificates From Firefox 58*. [Online]. Available: https://blog.mozilla.org/security/2017/08/30/removing-disabled-wosign-startcom-certificates-firefox-58/

[29] Y. Liu *et al.*, "An end-to-end measurement of certificate revocation in the Web's PKI," in *Proc. ACM Conf. Internet Meas. Conf. (IMC)*, 2015, pp. 183–196.

[30] Netcraft Ltd. (2018). *CRL Performance Monitoring*. [Online]. Available: http://uptime.netcraft.com/perf/reports/performance/CRL

[31] L. Zhu, J. Amann, and J. Heidemann, "Measuring the latency and pervasiveness of TLS certificate revocation," in *Proc. Int. Conf. Passive Active Netw. Meas.*, 2016, pp. 16–29.

[32] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide scanning and its security applications," in *Proc. 22nd USENIX Secur.*, 2013, pp. 605–620.

[33] Q. Hu, M. R. Asghar, and N. Brownlee, "Certificate revocation guard (CRG): An efficient mechanism for checking certificate revocation," in *Proc. IEEE 41st Conf. Local Comput. Netw. (LCN)*, Nov. 2016, pp. 527–530.

[34] A. Bates, J. Pletcher, T. Nichols, B. Hollembaek, and K. R. B. Butler, "Forced perspectives: Evaluating an SSL trust enhancement at scale," in *Proc. Conf. Internet Meas. Conf. (IMC)*, 2014, pp. 503–510.

[35] L. Dickinson, "Certificate revocation table: Leveraging locality of reference in Web requests to improve TLS certificate revocation," M.S. thesis, Dept. Comput. Sci., Brigham Young Univ., Provo, UT, USA, 2018.

[36] B. Amann, R. Sommer, M. Vallentin, and S. Hall, "No attack necessary: The surprising dynamics of SSL trust relationships," in *Proc. 29th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, 2013, pp. 179–188.

[37] R. Holz, L. Braun, N. Kammenhuber, and G. Carle, "The SSL landscape: A thorough analysis of the X. 509 PKI using active and passive measurements," in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf. (IMC)*, 2011, pp. 427–444.

[38] D. Sgandurra and E. Lupu, "Evolution of attacks, threat models, and solutions for virtualized systems," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–38, Feb. 2016.

[39] R. Patil and C. Modi, "An exhaustive survey on security concerns and solutions at different components of virtualization," *ACM Comput Surv.*, vol. 52, no. 1, pp. 1–38, 2019.

[40] J. Rutkowska and R. Wojtczuk, "Preventing and detecting Xen hypervisor subversions," in *Proc. Blackhat Briefings USA*, 2008, p. 119.

[41] J. Wang, A. Stavrou, and A. Ghosh, "HyperCheck: A hardware-assisted integrity monitor," in *Proc. Int. Workshop Recent Adv. Intrusion Detection*, 2010, pp. 158–177.

[42] A. M. Azab, P. Ning, Z. Wang, X. Jiang, X. Zhang, and N. C. Skalsky, "HyperSentry: Enabling stealthy in-context measurement of hypervisor integrity," in *Proc. 17th ACM Conf. Comput. Commun. Secur. (CCS)*, 2010, pp. 38–49.

[43] Z. Wang and X. Jiang, "HyperSafe: A lightweight approach to provide lifetime hypervisor control-flow integrity," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 380–395.

[44] B. Ding, Y. He, Y. Wu, and Y. Lin, "HyperVerify: A VM-assisted architecture for monitoring hypervisor non-control data," in *Proc. IEEE 7th Int. Conf. Softw. Secur. Rel. Companion*, Jun. 2013, pp. 26–34.

[45] J. Szefer, E. Keller, R. B. Lee, and J. Rexford, "Eliminating the hypervisor attack surface for a more secure cloud," in *Proc. 18th ACM Conf. Comput. Commun. Secur. (CCS)*, 2011, pp. 401–412.

[46] X. Wang, Y. Qi, Y. Dai, Y. Shi, J. Ren, and Y. Xuan, "TrustOSV: Building trustworthy executing environment with commodity hardware for a safe cloud," *J. Comput.*, vol. 9, no. 10, pp. 2303–2314, Oct. 2014.

[47] A. Nguyen, H. Raj, S. Rayanchu, S. Saroiu, and A. Wolman, "Delusional boot: Securing hypervisors without massive re-engineering," in *Proc. 7th ACM Eur. Conf. Comput. Syst. (EuroSys)*, 2012, pp. 141–154.

[48] W. Pan, Y. Zhang, M. Yu, and J. Jing, "Improving virtualization security by splitting hypervisor into smaller components," in *Proc. 26th IFIP DBSec*, 2012, pp. 298–313.

[49] Z. Wang, C. Wu, M. Grace, and X. Jiang, "Isolating commodity hosted hypervisors with hyperlock," in *Proc. 7th ACM Eur. Conf. Comput. Syst. (EuroSys)*, 2012, pp. 127–140.

[50] C. Wu, Z. Wang, and X. Jiang, "Taming hosted hypervisors with (mostly) deprivileged execution," in *Proc. 22nd NDSS*, 2013, pp. 1–15.

[51] L. Shi *et al.*, "Deconstructing Xen," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2017, pp. 1–15.

[52] R. Chandramouli, "Security recommendations for hypervisor deployment," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 800-125A, 2014.

[53] J. Ren, L. Liu, D. Zhang, Q. Zhang, and H. Ba, "Tenants attested trusted cloud service," in *Proc. IEEE 9th Int. Conf. Cloud Comput. (CLOUD)*, Jun. 2016, pp. 600–607.

[54] J. Schiffman, T. Moyer, H. Vijayakumar, T. Jaeger, and P. McDaniel, "Seeding clouds with trust anchors," in *Proc. ACM workshop Cloud Comput. Secur. Workshop (CCSW)*, 2010, pp. 43–46.

[55] R. Russell, "Virtio: Towards a de-facto standard for virtual I/O devices," *ACM SIGOPS Operating Syst. Rev.*, vol. 42, no. 5, pp. 95–103, Jul. 2008.

[56] A. Bates *et al.*, "Securing SSL certificate verification through dynamic linking," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2014, pp. 394–405.

[57] S. Fahl, M. Harbach, H. Perl, M. Koetter, and M. Smith, "Rethinking SSL development in an appified world," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 49–60.

[58] M. O'Neill *et al.*, "TrustBase: An architecture to repair and strengthen certificate-based authentication," in *Proc. 26th USENIX Secur.*, 2017, pp. 609–624.
[59] P. Szalachowski, L. Chuat, T. Lee, and A. Perrig, "RITM: Revocation in the middle," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2016, pp. 189–200.
[60] D. Chu *et al.*, "Secure cryptography infrastructures in the cloud," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–7.
[61] T. Freeman *et al.*, *Server-Based Certificate Validation Protocol (SCVP)*, document RFC 5055, 2007.
[62] *Group Policy Processing and Precedence*, Microsoft Corp., Redmond, WA, USA, 2006.
[63] T. Smith, L. Dickinson, and K. Seamons, "Let's revoke: Scalable global certificate revocation," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–14.
[64] The Chromium Projects. (2015). *CRLSets*. [Online]. Available: https://dev.chromium.org/Home/chromium-security/crlsets
[65] P. Eckersley and J. Burns, "An observatory for the SSLiverse," in *Proc. DEFCON*, 2010, p. 47.
[66] B. VanderSloot, J. Amann, M. Bernhard, Z. Durumeric, M. Bailey, and J. A. Halderman, "Towards a complete view of the certificate ecosystem," in *Proc. ACM Internet Meas. Conf. (IMC)*, 2016, pp. 543–549.
[67] N. Vallina-Rodriguez, J. Amann, C. Kreibich, N. Weaver, and V. Paxson, "A tangled mass: The Android root certificate stores," in *Proc. 10th ACM Int. Conf. Emerg. Netw. Exp. Technol. (CoNEXT)*, 2014, pp. 141–148.
[68] SophosLabs. (2010). *Who's Your Verisign-Malware Faking Digital Signatures*. [Online]. Available: https://nakedsecurity.sophos.com/2010/06/23/trojbhoqp-verisign
[69] L. Zeltser. (2018). *How Digital Certificates are Used and Misused*. [Online]. Available: https://zeltser.com/how-digital-certificates-are-used-and-misused/
[70] A. Alsaid and C. Mitchell, "Installing fake root keys in a PC," in *Proc. 2nd EuroPKI*, 2005, pp. 227–239.
[71] X. de Carné de Carnavalet and M. Mannan, "Killed by proxy: Analyzing client-end TLS interception software," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2016, pp. 1–17.
[72] Z. Durumeric *et al.*, "The security impact of HTTPS interception," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2017, pp. 1–14.
[73] J. Braun, F. Volk, J. Classen, J. Buchmann, and M. Mühlhäuser, "CA trust management for the Web PKI," *J. Comput. Secur.*, vol. 22, no. 6, pp. 913–959, Dec. 2014.
[74] M. Goodwin. (2015). *Revoking Intermediate Certificates: Introducing OneCRL*. [Online]. Available: https://blog.mozilla.org/security/2015/03/03/revoking-intermediate
[75] A. Schulman, D. Levin, and N. Spring, "RevCast: Fast, private certificate revocation over FM radio," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2014, pp. 779–810.
[76] J. Kasten, E. Wustrow, and J. A. Halderman, "Cage: Taming certificate authorities by inferring restricted scopes," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2013, pp. 329–337.
[77] C. Soghoian and S. Stamm, "Certified lies: Detecting and defeating government interception attacks against SSL," in *Proc. 15th FC*, 2011, pp. 250–259.
[78] D. Wendlandt, D. G. Andersen, and A. Perrig, "Perspectives: Improving SSH-style host authentication with multi-path probing," in *Proc. USENIX Annu. Tech. Conf.*, 2008, pp. 321–334.
[79] P. Szalachowski, S. Matsumoto, and A. Perrig, "PoliCert: Secure and flexible TLS certificate management," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2014, pp. 406–417.
[80] B. Li, W. Wang, L. Meng, J. Lin, X. Liu, and C. Wang, "Elaphurus: Ensemble defense against fraudulent certificates in TLS," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2019, pp. 246–259.
[81] S. Fahl, M. Harbach, T. Muders, M. Smith, L. Baumgärtner, and B. Freisleben, "Why eve and mallory love Android: An analysis of Android SSL (in) security," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2012, pp. 50–61.
[82] M. Georgiev, S. Iyengar, S. Jana, R. Anubhai, D. Boneh, and V. Shmatikov, "The most dangerous code in the world: Validating SSL certificates in non-browser software," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2012, pp. 38–49.
[83] B. He *et al.*, "Vetting SSL usage in applications with SSLINT," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 519–534.
[84] D. Sounthiraraj, J. Sahs, G. Greenwood, Z. Lin, and L. Khan, "SMV-HUNTER: Large scale, automated detection of SSL/TLS man-in-the-middle vulnerabilities in Android apps," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2014.
[85] C. Brubaker, S. Jana, B. Ray, S. Khurshid, and V. Shmatikov, "Using frankencerts for automated adversarial testing of certificate validation in SSL/TLS implementations," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 114–129.
[86] K. Krombholz, W. Mayer, M. Schmiedecker, and E. Weippl, "I have no idea what I'm doing—On the usability of deploying HTTPS," in *Proc. 26th USENIX Secur.*, 2017, pp. 1339–1356.
[87] N. Vratonjic, J. Freudiger, V. Bindschaedler, and J.-P. Hubaux, "The inconvenient truth about Web certificates," in *Proc. Workshop Econ. Inf. Secur. (WEIS)*, 2011, pp. 1–38.
[88] R. Holz, J. Amann, O. Mehani, M. Wachs, and M. A. Kaafar, "TLS in the wild: An Internet-wide analysis of TLS-based protocols for electronic communication," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2016, pp. 1–15.

Bingyu Li received the B.S. degree from Jilin University in 2013 and the Ph.D. degree from the University of Chinese Academy of Sciences in 2020.

He holds a postdoctoral position at the School of Cyber Science and Technology, Beihang University. His research interests include public key infrastructure, identity management, applied cryptography, and system security.

Jingqiang Lin (Senior Member, IEEE) received the M.S. and Ph.D. degrees from the University of Chinese Academy of Sciences, in 2004 and 2009, respectively.

He is a Full Professor with the School of Cyber Security, University of Science and Technology of China. His research interests include applied cryptography and system security.

Qiongxiao Wang received the B.E. degree from Jilin University and the Ph.D. degree from the University of Chinese Academy of Sciences. She is currently an Associate Professor with the Institute of Information Engineering, Chinese Academy of Sciences. Her research interests include applied cryptography and identity management.

Ze Wang received the Ph.D. degree from the University of Chinese Academy of Sciences in 2018. He is currently a Principal Engineer of Cyber Security with the Central Software Institute, Huawei. His research interests include applied cryptography, network security, and AI security.

Jiwu Jing (Member, IEEE) received the bachelor's degree from the Department of Electronics Engineering, Tsinghua University, in 1987, and the M.Sc. and Ph.D. degrees from the Graduate School, University of Science and Technology of China, in 1990 and 2003, respectively.

He was the Convenor of the Expert Group on Information Security of the National High Technology Research and Development Program of China (863 Program). He is a Full Professor with the School of Computer Science and Technology, University of Chinese Academy of Sciences. His main research interests include public key infrastructure, identity management, fault tolerance, mobile security, information systems, and network protection. He received the National Science and Technology Progress Award, the Science and Technology Progress Award of the Chinese Ministry of Electronics Industry, and the Science and Technology Progress Award of CAS.